9-9-2007

Optimal digital filter design for dispersed signal equalization

Andrew Mihalik

Candidate

Department

This thesis is approved, and it is acceptable in quality and form for publication on microfilm:

Approved by the Thesis Committee:

, Chairperson

Accepted:

Dean, Graduate School

Date

Optimal Digital Filter Design for Dispersed Signal Equalization

BY

Andrew E. Mihalik

B.S., Electrical Engineering, University of New Mexico, 2002

THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science
Electrical Engineering

The University of New Mexico

Albuquerque, New Mexico

July, 2007

ACKNOWLEDGMENTS

I would like to thank my advisor and thesis chair, Dr. Marios Pattichis, for his advice, and most of all patience, in helping me through this challenging thesis. The optimistic outlook and can-do attitude he displays were a great encouragement to me. His dedication to his work and his students is unparalleled. From him I learned the importance of perseverance and staying focused.

I also thank my other committee members, Dr. Christos Christodoulou and Dr. Armin Doerry, for their willingness to review this manuscript and the open-door policy they have maintained for the students in the department. From Dr. Christodoulou I learned to dissect a difficult task into manageable pieces. It is easy to be overwhelmed by the complexity of the things we study.

Dr. Doerry has served in a mentorship role for many students in the department. From him I learned to temper the mathematical treatment of signal processing with common sense. His selfless approach to life has earned him the respect of everyone who works with him. I think he is a role model of how to be a good engineer, but more importantly how to be a good person. Thank you for volunteering your time and always going out of your way to help people.

My gratitude goes out to Mr. William Breiland. The clarity of his analytical thought and problem-solving capabilities are really extraordinary. He introduced me to the topic of this thesis, and I am thankful for the many beneficial discussions we had. His thorough treatment of subjects made working with him a pleasure.

Let me thank my family also, for their confidence that I could succeed. Their "you can do it" mantra gave me the support I needed.

Optimal Digital Filter Design for Dispersed Signal Equalization

BY

Andrew E. Mihalik

ABSTRACT OF THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science
Electrical Engineering

The University of New Mexico

Albuquerque, New Mexico

July, 2007

Optimal Digital Filter Design for Dispersed Signal Equalization

BY

Andrew E. Mihalik

B.S., Electrical Engineering, University of New Mexico, 2002

M.S., Electrical Engineering, University of New Mexico, 2007

Abstract

Any signal a satellite receives from Earth has traveled through the ionosphere. Transmission through the ionosphere results in a frequency dependent time-delay of the signal frequency components. This effect of the medium on the signal is termed dispersion, and it increases the difficulty of pulse detection. A system capable of compensating for the dispersion would be desirable, as pulsed signals would be more

readily detected after compression. In this thesis, we investigate the derivation of a digital filter to compensate for the dispersion caused by the ionosphere.

A transfer function model for the analysis of the ionosphere as a system is introduced. Based on the signal model, a matched filter response is derived. The problem is formulated as a group delay compensation effort. The Abel-Smith algorithm is employed for the synthesis of a cascaded allpass filter bank with desired group delay characteristics. Extending this work, an optimized allpass filter is then derived using a pole location approach. A mean-square error metric shows that the optimized filter can reproduce, and even improve upon, the results of the Abel-Smith design with a significantly lower order filter. When compared against digital filters produced with the least p-th minimax algorithm, we find that the new method exhibits significantly lower error in the band of interest, as well as lower mean squared error overall. The result is a simple optimized equalization filter that is stable, robust against cascading difficulties, and applicable to arbitrary waveforms. This filter is the cornerstone to a new all-digital electromagnetic pulse detection system.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Satellites have been monitoring for electromagnetic pulses (EMPs) originating from Earth for decades. The environment presented by a satellite is challenging from a system design viewpoint. Power consumption is a critical factor, as well as the fact that considerable radiation may be present. Space is at a premium, and weight must be minimized. Analog circuits exist for detecting these pulses, which may originate from lightning or artificial pulses. However, these circuits suffer from the inherent disadvantages of analog systems: drift (very slow changes in the system over time), lack of changeability, component degradation, etc.

It may be possible to create an all-digital pulse compression filter, due to the substantial capability digital hardware possesses today. Digital systems are more immune to component aging and drift is not a factor. Signal attenuation and interference are also eliminated. The power requirements on a satellite can be very restrictive, making the efficiency of digital hardware important. Digital systems are more easily reconfigured, so alterations and improvements may be implemented without resorting to new hardware. The development of a digital anti-dispersion filter would be a critical element in forming a more capable EMP detector.

1.1 Motivation

The predominant signal processing issue to consider in a satellite based pulse detection system is the fact that the pulse must travel through the ionosphere. The ionosphere acts as a dispersive medium to EM signals that interact with it [3]. This means the energy in the pulse is spread out in time; the pulse becomes "smeared out". According to the conservation of energy principle, the signal magnitude must drop as the waveform is spread out in time. This smearing makes the pulse more difficult to detect compared to the original sharp pulse [13]. If a signal processing technique could effectively reverse the dispersion caused by the ionosphere, it would be a large step towards improving the likelihood of pulse detection.

A review of the requirements yielded an approximate constraint on the order of the filter. The order of the filter is associated with the number of operations required by the algorithm. It is this constraint on the order of the filter that motivated the derivation of a filter using optimization techniques.

1.2 Thesis Statement, Innovations, and Contributions

In this thesis, I will show that ionosphere pulse dispersion can be compensated for using an optimal group delay matching process. The group delay matching process used is robust against the difficulties usually encountered with high-order cascaded systems. Robustness is achieved by ensuring the optimal filter maintains the allpass characteristic. This is important because the ultimate filter is composed of multiple stages to account for the varying ionic density of the ionosphere. Also, the technique employed to derive the

equalization filter is general enough that it does not hinge on properties of the ionosphere system model. The adaptability of this filter derivation ensures it will remain applicable even when more advanced ionosphere models are available. The optimization method developed is demonstrated to be convergent even with random initializations. The method also is very general in its formulation; it is able to synthesize an arbitrary group delay waveform.

Several innovations are included in this thesis. Originally, the Abel-Smith design was created for use in music synthesis. The application of the Abel-Smith group delay matching technique in ionosphere pulse compression is new. Also, the optimization of the group delay matching algorithm is a new extension to the Abel-Smith method. Although it separated the frequency band of interest into subbands, the Abel-Smith method did not attempt to match these subbands in a separately customized manner. The optimization routine takes advantage of these subbands more effectively. As a result, the optimization extension creates filters that more accurately match the group delay, compared to the Abel-Smith approach for a given filter order. For instance, for an 8^{th} order filter a 79% reduction of the mean-squared error was realized. For the 20^{th} order filter case, the reduction was nearly 98%. The reduction of the filter order for a given error specification yields a large reduction in the computational expense, which is critical to the feasibility of the whole system. When compared against digital filters produced with the least p-th minimax algorithm, we find that the new method exhibits significantly lower error in the band of interest, as well as lower mean squared error overall. Another

attractive feature of the optimization approach is the simplicity of its design, based solely on pole locations.

1.3 Thesis Summary

Chapter 1 of this thesis has introduced the problem of ionosphere dispersion, and the approach we chose to counteract it. It highlights the fact that our design is destined for an all-digital detection system.

In Chapter 2, we will discuss some background pertinent to the subject, and introduce the notion of the total electron content of the ionosphere. We will show how the total electron content is incorporated into a model of the ionosphere that captures its gross effects. Previous work on the subject is touched upon briefly. We use the Appleton-Lassen equation to develop a transfer function description of the ionosphere. By the end of the chapter, we have set the stage for a cascaded filter to undo the dispersion of the ionosphere.

Chapter 3 speaks about the implementation of an IIR anti-dispersion filter. A group delay matching approach is proposed. This formulates the problem as an effort to create a filter that exhibits essentially an inverted phase response behavior compared to the ionosphere. The desire for a cascaded, prescribed group delay filter leads us into a description of allpass filters. These allpass filters serve as building blocks towards a biquad cascading method proposed by Abel and Smith. The Abel-Smith method is detailed, and by the end of the chapter we use optimization techniques to improve their

method. We use a trust region approach, and compare it to the Abel-Smith approach and also to a line search optimization method.

The results of the work we have done are contained in Chapter 4. We show that our method produces optimal, stable, allpass filters of arbitrary order regardless of initialization. We demonstrate that the filters are stable, and compare the errors produced for various filter orders. Also, we include overlaid plots demonstrating the group delay fit exhibited by each of the filters. Also, we explore how the magnitude of the error is reduced with higher filter order. Uniqueness of the optimal solution is suggested by multiple trial runs, verified with pole-zero plots. From a numerical perspective, we show that initializing the optimization routine with the Abel-Smith results vs. a random initialization is advantageous. To establish generality of our method, we include a filter group delay plot that closely matches an arbitrary waveform with an 8^{th} order filter. The end result is that the work contained in this thesis brings the idea of a digital ionosphere compensation filter a significant step closer to reality.

Chapter 5 brings our presentation to a close with some conclusions, and also the extensive future work that is warranted for this subject. We mention here the possibility of filters with complex coefficients, and speculate if abandoning the insistence on conjugate sections will add another degree of freedom to the optimization objective.

Chapter 2

Background

Near the ground the air is almost unionised and its electrical conductivity is negligibly small. The index of refraction of the air is almost equal to that of a vacuum. Although water vapor can contribute to refraction of a wave, usually it is assumed that the non-conducting atmosphere below the ionosphere is the same as free space. However, signals traveling from the atmosphere up to a satellite must pass through the ionosphere. The ionosphere certainly cannot be considered to be the same as free space. Above about 1000 km it is a relatively good conductor [2]. This has a very noticeable effect on the propagating wave: a frequency dependent delay that is termed dispersion.

2.1 Previous Work on Compression Filters

A notable effort was put forth in the FORTE (Fast On-orbit Recording of Transient Events) remote-sensing satellite application. This project was undertaken in the '90s by DARPA, the Information Sciences Institute, and Los Alamos National Laboratory. The purpose of the FORTE payload was to detect, record, and characterize lightning occurring in the earth's atmosphere. It consisted of an RF receiver and several sub-band trigger units that operate in several bands of the working bandwidth of 20 to 310 MHz. The design represented a significant advance in the technology for performing this task. The receiver is low noise, and the sub-band trigger units provided more immunity from

detecting unwanted carrier signals. Multiple trigger signals would have to be generated in separate bands for an ultimate detection trigger to cycle. It would be unlikely that a communications carrier signal would cause triggers in any more than one band. Since the signal of interest is impulsive in time, it has a broad frequency content that would cause triggers over multiple bands. The instrument allowed unprecedented inter-cloud lightning rate measurements to be made.

One detail to realize here is that the instrument did not actually perform the waveform compression in a digital system. The compression filter used was analog, and then the resulting compressed signal was digitized for further processing chains. The effort we are making here is to perform the compression in a digital manner for the aforementioned advantages.

The predecessor to the FORTE system was the Blackbeard instrument, completed in 1993. Blackbeard was an astrophysics and geophysics research satellite monitoring celestial X-rays and making broad-band VHF measurements. It was designed to study the distortion and interference effects on transient trans-ionospheric VHF signals such as lightning and artificial pulses. Most notably it observed strange radio bursts called Trans-Ionospheric Pulse Pairs, or TIPPs. In 1996, scientists reported the first simultaneous Blackbeard observations and multiple ground station measurements of TIPPs. This was taken as evidence suggesting that TIPPs come from thunderstorms and probably comprise one atmospheric event and its reflection off Earth. A main topic of study for

the experiment was the refinement of ionospheric transfer function models. Blackbeard, too, did not use a digital system for the compression filter.

2.2 Ionosphere Dispersion

The ionosphere contains nearly free electrons, with the level of ionization maintained by ultra-violet light from the Sun (which effectively knocks electrons out of neutral atoms). In general, the refractive index, n, of a gaseous medium is a function of the wave frequency, f [1]. Since the effective wave speed through the medium is c/n, with c equal to the speed of light, it follows that waves of different frequencies traveling through the ionosphere do so at *different* speeds. Higher frequency components of the signal propagate faster, and thus arrive earlier, than lower frequency components. The received signal power is reduced, as the energy contained in the signal is spread over a greater length of time. This dispersion of the signal is an important factor in any signal analysis dealing with the ionosphere. To improve detectability of the received signal by increasing peak power, the signal content must be compressed in time. We will often refer to the un-dispersion process as "compression", in the sense that the signal's temporal extent is compressed.

The total electron content (TEC) is a measure of the total amount of electrons along a particular line of sight through the ionosphere. The TEC value can be considered to characterize the gross effect of the ionosphere on an electromagnetic signal. The TEC varies with location, solar exposure and solar activity. Since the TEC varies with different factors, the behavior of the ionosphere is by no means a constant with respect to

geographic location, annual seasons, or time of day. The TEC is not a value that can be predicted easily, and it is practical to assume that its value is unknown. The TEC is measured in TECU, or Total Electron Content Units. The amount of dispersion is dependent on the TEC value, as will be discussed shortly.

A detection scheme for EMPs therefore should take into account the dispersive behavior of the medium. A matched filter can be used to “undo” the effect of the ionosphere, in the sense that it applies the inverse of the phase shifting behavior that originally distorted the signal. The phase shifting behavior is a consequence of the frequency dependent time delays induced by the ionosphere. Proper matched filtering shrinks the time duration of the signal, without affecting the energy content. Thus the instantaneous signal power increases and the probability of detection is improved. The signal pulse, after the dispersion has taken place, appears similar to a frequency modulated radar pulse. This type of radar pulse is often called a chirped pulse, with the implementation of the chirped pulse in the radar system termed pulse compression. In general, a longer pulse allows more energy to be emitted, and hence received, but usually hinders range resolution. But in a chirped radar, this longer pulse also has a frequency shift during the pulse (hence the chirp or frequency ramping effect). When the "chirped" signal is returned, it must be correlated with the sent pulse. Classically, in analog systems, it is passed to a dispersive delay line (often a Surface Acoustic Wave device) that has the property of varying velocity of propagation based on frequency. This technique "compresses" the pulse in time - thus having the effect of a much shorter pulse (improved range resolution) while having the benefit of longer pulse length (much more

signal returned). Analog matched filters are heavy and less than compact. Usually, they have been designed to work with a pulse whose frequency is ramping up in a linear fashion. As will be discussed shortly, the ionosphere does not disperse the signal in a linear fashion. The extensive work that has been performed creating linear chirp matched filters would not be directly useable in compressing the EMP.

2.3 Ionosphere Transfer Function

As mentioned, the TEC is one way to characterize the state of the ionosphere and how it will affect signals passing through it. A simple model that captures the gross effects of the ionospheric dispersion is considered. This model idealizes the ionosphere by assuming that all components of volume are the same in composition and EM properties (homogeneous slab). The underlying equation that governs the ionosphere model is a form of the Appleton-Lassen equation [12]. The modified Appleton-Lassen equation yields the refractive index of the ionosphere as a function of the TEC, and will be given shortly. Since the ionosphere is a dispersive medium, its index of refraction varies with different frequencies of electromagnetic radiation.

A wave traveling through a distance X (the thickness of the ionosphere slab) will take a certain amount of time to travel the distance, given it is traveling at c (the speed of light in vacuum). If we ignore for a moment that the wave is traveling through the ionosphere, and we employ the familiar expression time=distance/rate, then

$$\tau_{vac} = X / c. \tag{2.1}$$

Equation (2.1) yields τ_{vac}, the vacuum time delay that would be experienced by an EM signal traversing the distance X. For an EM signal in a medium, the group velocity is given by

$$v_g = nc, \tag{2.2}$$

where n is the refractive index of the medium. Then, again using the expression time=distance/rate,

$$time = X / v_g = X /(nc) = (X / c) / n. \tag{2.3}$$

This means if we consider the fact that the wave is, in fact, traveling through the ionosphere, not through vacuum, the delay time is

$$\tau_f = X /(n_f c) = \tau_{vac} / n_f . \tag{2.4}$$

This expression for the frequency dependent delay time shows that the relationship between the vacuum delay time and the actual delay time is governed by the ionosphere refractive index n_f.

For a pulse that contains significant frequencies from f_{lo} to f_{hi}, components near f_{hi} arrive at $\tau_{vac} / n_{f_{hi}}$ and components near f_{lo} arrive at $\tau_{vac} / n_{f_{lo}}$, where n_f is given by a form of the Appleton-Lassen equation [13]:

$$n_f = \sqrt{1 - aTEC / f^2}. \tag{2.5}$$

In (2.5), *TEC* is the TEC value in TECU, where 1 TECU=10^{16} electrons/m^2. The parameter a is a constant that is dependent on the ionosphere slab thickness:

$$a = \frac{q_e^{\,2}}{4\pi^2 \varepsilon_0 m_e X} = 2.0154096 \text{ MHz}^2. \tag{2.6}$$

In (2.6), q_e is the charge of an electron, m_e is the mass of an electron, ε_0 is the permittivity of a vacuum, and X is the thickness of the ionosphere. Based on the TEC value, a description of the transfer function of the ionosphere is given [13]:

$$H(f) = \exp[-j2\pi f(n_f \tau_{vac} - \tau_{mid})]. \tag{2.7}$$

Here, the midway (or average) delay time is given by

$$\tau_{mid} = \frac{1}{2}\left(\frac{\tau_{vac}}{n_{f_{hi}}} + \frac{\tau_{vac}}{n_{f_{lo}}}\right). \tag{2.8}$$

Numerically, the vacuum delay time is computed using a value of X = 400km, yielding

$$\tau_{vac} = X / c = 1334.25638\ \mu s. \tag{2.9}$$

The values are given for frequency measured in MHz, and TEC in TECU.

We can rewrite (2.5) as

$$n_f = (1 - aTEC / f^2)^{1/2}\ , \tag{2.10}$$

enabling us to recognize that (2.10) is in a form conducive to application of the binomial theorem. As a consequence of the binomial theorem, the following power series can be derived:

$$(1+x)^p = 1 + px + p(p-1)x^2/2 + p(p-1)(p-2)x^3/6 + \cdots, \quad |x| < 1. \tag{2.11}$$

For high frequencies or low TEC values, an approximation can be made to the expression for the refractive index using the first two terms from the power series. From the binomial series, substitute $x = -aTEC / f^2$ and $p = 1/2$ into (2.11) to obtain:

$$n_f \cong 1 - aTEC/(2f^2). \tag{2.12}$$

This leads to the updated expression for the term $n_f \tau_{vac} - \tau_{mid}$ from (2.7), using the approximation $1/(1-x) = 1 + x$ for small x:

$$
\begin{aligned}
n_f \tau_{vac} - \tau_{mid} &\cong \tau_{vac}\left(1 - \frac{aTEC}{2f^2} - \frac{1}{2}\left(2 + \frac{aTEC}{2f_{hi}{}^2} + \frac{aTEC}{2f_{lo}{}^2}\right)\right) \\
n_f \tau_{vac} - \tau_{mid} &\cong -\frac{aTEC\,\tau_{vac}}{2}\left(1/f^2 + 1/F^2\right), \qquad (2.13) \\
\text{where } 1/F^2 &= \frac{1}{2}\left(1/f_{hi}{}^2 + 1/f_{lo}{}^2\right).
\end{aligned}
$$

This carries through to the approximation for the ionosphere transfer function:

$$
\begin{aligned}
H_{iono}(f) &\cong e^{j2\pi bfTEC(1/F^2 + 1/f^2)} \\
&where\ b = \frac{q_e{}^2}{8\pi^2 \varepsilon_0 m_e c} = 1344.5 \text{ MHz}. \qquad (2.14)
\end{aligned}
$$

Note that the magnitude of the transfer function is unity in (2.13), and the transfer function is described completely as a phase term:

$$
\begin{aligned}
H_{iono}(f) &\cong e^{j2\pi f b\mathrm{TEC}\left(1/F^2 + 1/f^2\right)} \\
&= e^{j2\pi f b\mathrm{TEC}/F^2} e^{j2\pi b\mathrm{TEC}/f}. \qquad (2.15)
\end{aligned}
$$

We can see that the first exponential term in the final line of (2.15) fits into the form of the Time Delay Theorem for the Fourier transform. The Time Delay Theorem for the Fourier transform states that

$$
\Im[h(t - t_0)] = e^{-j2\pi f t_0} H(f). \qquad (2.16)
$$

Equation (2.16) states that a time delay corresponds to a linear frequency phase shift. When we compare (2.15) to (2.16), we see that $t_0 = -bTEC/F^2$. Using this information in view of (2.15), it is clear that the $1/F^2$ term in the phase contributes only a constant time delay, and the remaining $1/f$ term is responsible for the dispersion. The equation of the refractive index, n_f, in (2.5) shows that for frequency $f_p = \sqrt{aTEC}$, the refractive index is zero. This value of frequency is termed the plasma frequency, and is the limiting

frequency at or below which a radio wave is reflected by the ionosphere at vertical incidence. If the transmitted frequency is higher than the plasma frequency of the ionosphere, then the EM energy travels through the ionosphere. It is calculated as shown below:

$$f_p^2 = \frac{\mathrm{TEC} \cdot q_e^2}{4\pi^2 \varepsilon_0 m_e X} = 2.0154096 \cdot \mathrm{TEC\ MHz}^2. \tag{2.17}$$

The main idea to take from this section is that the ionosphere transfer function we are using in the analysis that follows in the next chapter is

$$H_{iono}(f) \cong e^{j2\pi b \mathrm{TEC}/f}. \tag{2.18}$$

2.4 Description of the Signal Model

Let us consider the input signal that we are dealing with in this discussion of the ionosphere dispersion. As mentioned, the type of signal we would be searching for is a pulsed signal. Specifically, the signal is one whose duration is very short, and whose amplitude is relatively large compared with the signal duration. An example signal is shown in Figure 2.1.

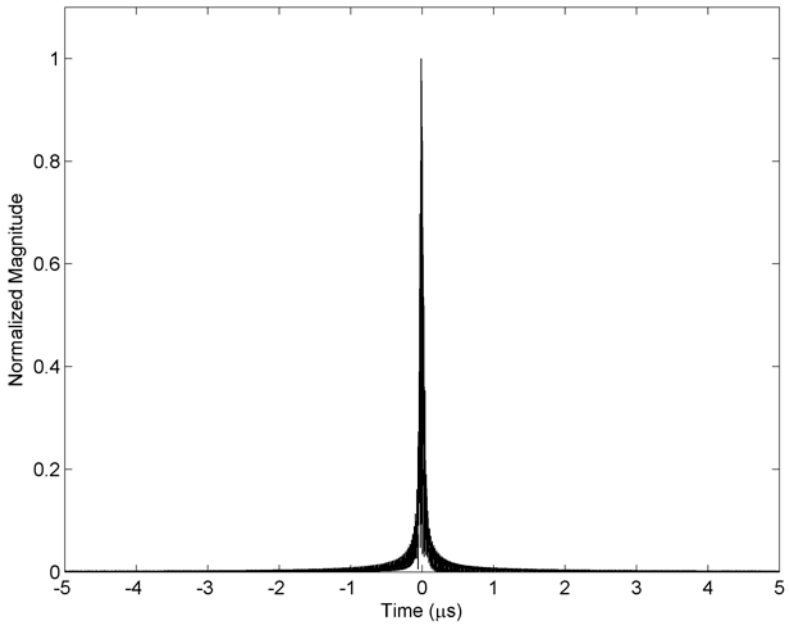


Figure 2.1 *Undispersed pulse.*

This signal was produced using the program code listed in Appendix A. The program code was based on the modified Appleton-Lassen equation. We can see from Figure 2.1 is that the signal is nearly impulsive in nature, and therefore has a fairly broad frequency content. Signals will not propagate through the ionosphere at frequencies below the plasma frequency; this effect, combined with the fact that the pulse signal is not infinitely sharp, reduces the practical range of RF frequencies to roughly 20 MHz to 150 MHz [13].

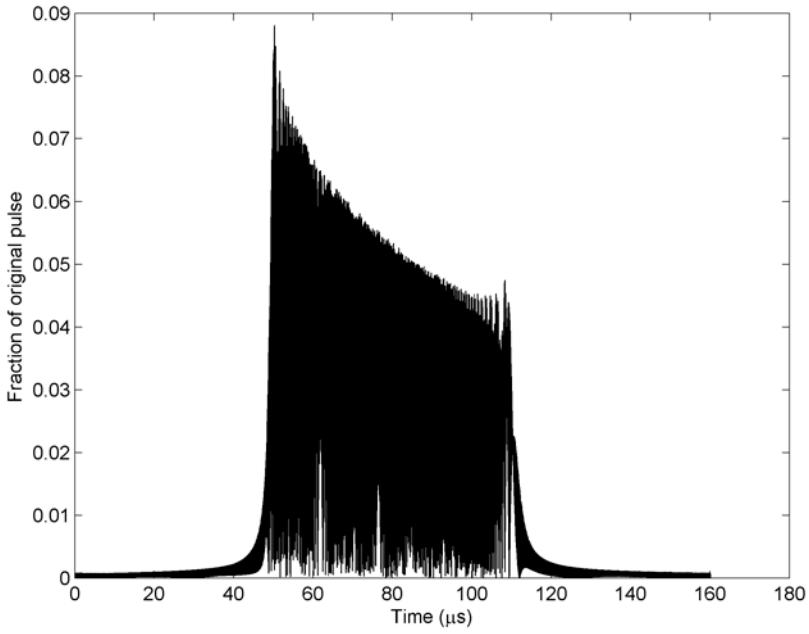


Figure 2.2 *Band-limited pulse magnitude TEC=100.*

Comparing Figure 2.2 to Figure 2.1, we can see how much wider the signal has spread out after traveling through the ionosphere. The dispersion has been simulated using the model discussed in the previous section. Figure 2.2 was created using the program code listed in Appendix A. The main fact to realize here is that a pulse signal that is transmitted through the ionosphere will not be received as that same pulse shape delayed by some constant transmission time. It will effectively be spread out over time, since the higher frequencies arrive before the lower frequencies. The spreading does, to a great degree, conserve the energy in the pulse. Therefore, the amplitude of the waveform is substantially reduced when we compare Figure 2.2 to Figure 2.1.

We will now give a brief description of the data collection setup. As mentioned, we are considering our signal of interest to be bandlimited. An analog anti-aliasing filter

precedes the digitizer, the bandwidth being limited to the interval 51.15 MHz to 76.625 MHz. These critical frequencies were chosen to be a fraction of the clock aboard the satellite. The sampling frequency is 51.15 MHz, which is also a fraction of the high-precision clock that is already available. Figure 2.3 shows a block diagram of the system.

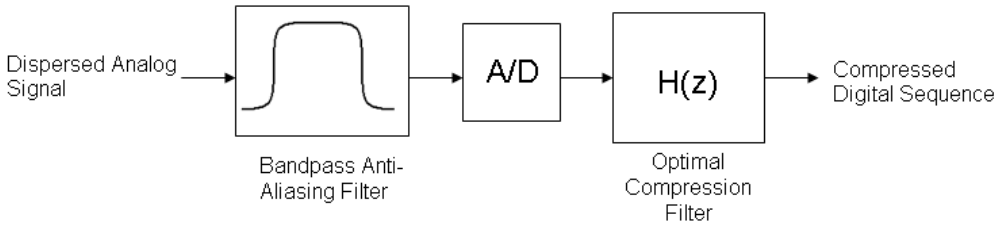


Figure 2.3 *Block diagram of compression scheme.*

Notice that the system is employing bandpass sampling [12], with the sampling rate set to twice the bandwidth. The original bandpass filtered spectrum is shown in Figure 2.4, and the spectrum after sampling is shown in Figure 2.5.

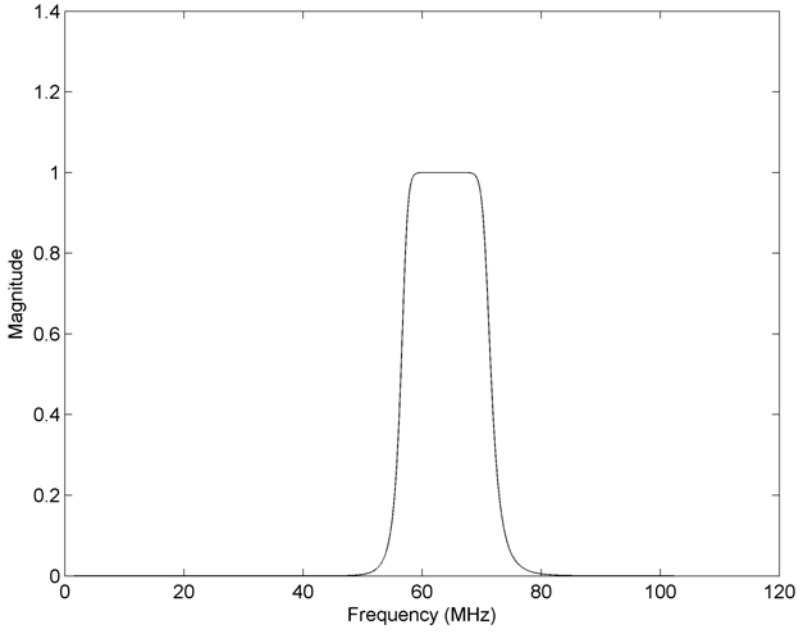


Figure 2.4 *Analog magnitude spectrum after anti-alias filtering.*

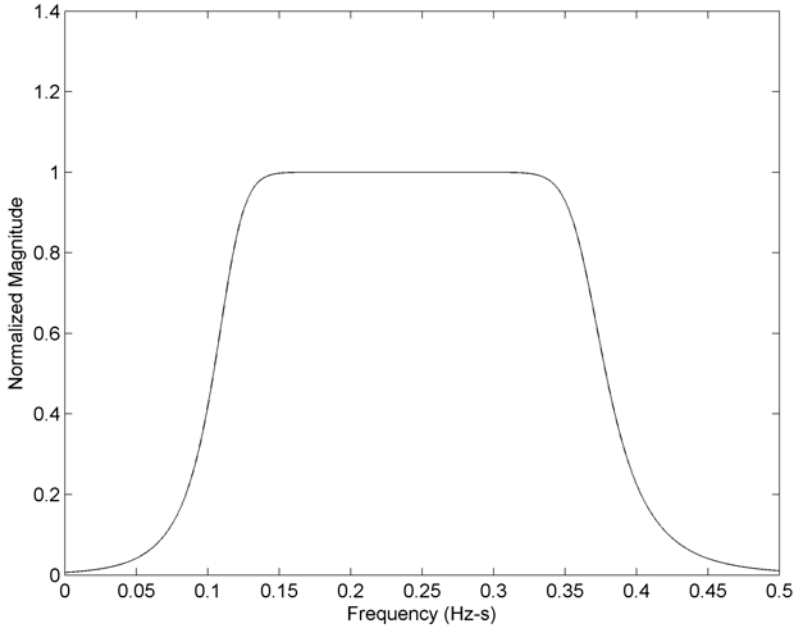


Figure 2.5 *Digital magnitude spectrum presented to compression filter.*

We can see in Figure 2.5 that a single period of the digital spectrum is shown. The plot is actually discrete but contains so many points that representation in an apparently discrete manner is not possible. We have used the bandpass sampling technique in such a manner that a copy of the original spectrum lies between DC and the signal bandwidth.

Now that we have described the signal we are dealing with, we are ready to discuss some approaches to counteract the effect of the dispersion.

2.5 Phase-Conjugate Filter

As shown in the previously, the transfer function of the ionosphere acts upon the phase of the signal according to (2.18). The ideal filter for undoing the effect the ionosphere would thus be tasked with correcting the phase shifting described by (2.18). The complex conjugate of the ionosphere transfer function yields the proper non-causal phase-conjugate filter:

$$H_{iono}(f)H_{iono}^{*}(f)=1. \tag{2.19}$$

The angle of the resulting realizable composite transfer function will result in a linear phase characteristic. Therefore the composite transfer function has no effect on the phase of the input signal; the phase distortion due to the ionosphere has been effectively canceled out. Clearly, the phase-conjugate filter approach is dependent on knowledge of the TEC value of the ionosphere. The TEC value is not, in general, a parameter that is known or stable. The EMP source, and receiving satellite, may be located anywhere in the world. One cannot simply build a phase-conjugate filter, since the phase function is not known. An error of 1 TECU yields a phase-conjugate filter output amplitude that is

less than half the properly matched response. If we knew the effective TEC value, we could build a filter that is suited to it, and accomplish the compression more readily. The problem to overcome, then, is to build a filter that does not require knowledge of the TEC value.

2.6 Cascade Filter vs. Parallel Filter

There are a myriad ways to deal with the unknown TEC factor. Two basic approaches might be to use either a parallel array of filters, or a cascade array of filters. Upon first inspection, a parallel system seems a more intuitive solution to the unknown TEC issue. A discussion of a parallel filter approach is appropriate before the advantages of the cascade system are introduced.

2.6.1 Parallel Filter

One way to solve the problem of the unknown TEC value is very direct. A number of phase-conjugate filters can be constructed, each corresponding to a different TEC value. Regardless of the actual TEC value, one of the filters would produce a large output with a dispersed pulse input. Because the matching of the filter is so dependent on the TEC value, small steps must be taken in TEC value between different filters. Since the filter which will produce the max output is unknown, each output would then undergo a thresholding procedure. The thresholding procedure is based on the amplitude of the compressed signal. Essentially, a more effective compression yields a higher amplitude of the pulse output. Based on the threshold test, a decision is made on the presence of an EMP, shown in Figure 2.6.

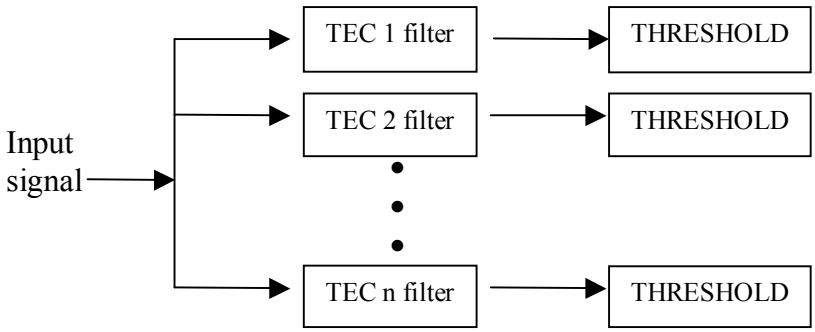


Figure 2.6 *Block diagram of parallel filter implementation.*

Fast convolution yields the individual responses from the filter bank. With possibly hundreds of different values of TEC, and considering the selective nature of the phase-conjugate filter, hundreds of different filter modules would need to be built. This results in increased complexity.

2.6.2 Cascade Filter

The complexity problem encountered in the parallel method of implementation is eliminated by the cascade method. Consider again the approximation for the ionosphere transfer function achieved with the binomial series, along with a rewritten form:

$$e^{-j2\pi bTEC/f} = \prod_{n=1}^{N} e^{-j2\pi b\alpha/f};\ \alpha = TEC/N \tag{2.20}$$

In other words, divide the exponent value by N, and then multiply N of these scaled terms, which will result in addition in the exponent N times. The block diagram for this situation is thus:

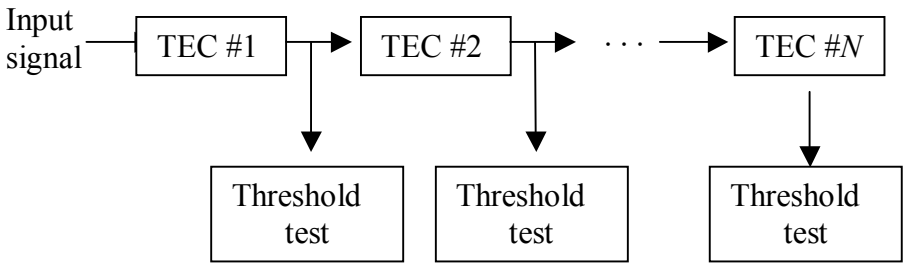


Figure 2.7 *Block diagram of cascade filter implementation.*

Each stage of the filter compresses the signal by an amount α. The transfer functions of the stages accumulate, so at the stage *p*, the compression is *p*α. To accommodate TEC values that range up to the value *N*, the number of stages necessary is also *N*. For example, suppose a waveform is input into a transfer function corresponding to the TEC of 50 TECU. In this case, the 50^{th} stage output of the filter will compress the waveform the optimum amount; the first 50 stages cascaded together form the phase-conjugate filter. As each new data point is fed in, any of the *N* filter stages can trip the threshold trigger. The larger delay associated with the cascade technique compared to the parallel technique is not detrimental to our goal. If the compressed signal comes out of the filter later in time, it is of no consequence to pulse detection. The results of the cascade produce the same compression as the parallel method, but in a simpler manner. It does not require building a separate filter for every TEC value, but rather iterating one compression stage multiple times. The use of this technique, however, requires the use of a cascade stage that will not accumulate error. This is important because the error in the stage is multiplied over and over as the signal is looped through. We will discuss such a filter in the next chapter.

CHAPTER 3

Methodology

A matched filter can be implemented when characteristics of the signal of interest are known. We will use the model we introduced in the previous chapter to develop a new formulation of the problem as a group delay fitting procedure. Often the non-linear phase distortion introduced by the use of an IIR filter is considered to be a harmful characteristic. Due to the non-linear phase response of the ionosphere, the phase distortion of an IIR can be considered a useful property. We need only be sure that the filter distortion counteracts the distortion of the ionosphere. The idea of an allpass filter is introduced, with a discussion of its properties that will be useful to our solution. The implementation of an IIR group delay optimization routine produces a low-order filter that effectively compresses the dispersed signal.

3.1 Group Delay Equalization of the Ionosphere Model

The phase response $\theta(f)$ of a filter gives the radian phase shift experienced by each sinusoidal component of the input signal. This measure of the phase response does not *directly* yield information about the frequency dependent time delays a signal will undergo when passed through the filter. Consider that if the input to a linear time-invariant filter is a sinusoid, the output of the filter is a sinusoid. If the input signal is $\sin(2\pi ft)$, then the output signal is $\sin(2\pi ft + \phi)$, where ϕ is the amount the sine wave

was shifted (as an angular measurement). The question remains, how much was the signal shifted in time? We know that the sine function is zero when its argument is zero. For the original sine wave, this must occur at time $t_0 = 0$ (since we know that the frequency f is not zero). From Figure 2.1, we can see the output reaches the same point in the waveform at time t_d.

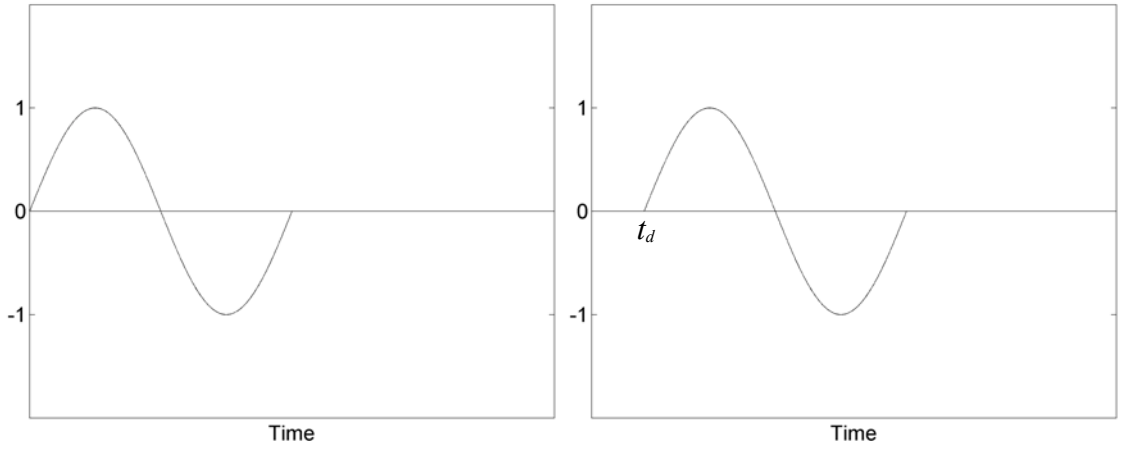


Figure 3.1 *Original and delayed sine waveforms.*

Since the argument of the shifted waveform is still zero at this crossing point, we can state:

$$2\pi f t_d + \phi = 0. \tag{3.1}$$

Solving for t_d, we obtain:

$$t_d = \frac{-\phi}{2\pi f}. \tag{3.2}$$

If we realize that for a given phase response, the phase is actually a function of the frequency, we arrive at a more intuitive measure called the phase delay, defined as

$$\tau_p(f) \equiv -\frac{\theta(f)}{2\pi f}. \tag{3.3}$$

The phase delay gives the time delay in seconds experienced by each sinusoidal component of the input signal.

A common measure derived from the filter phase response is called the group delay, defined by

$$\tau_g(f) \equiv -\frac{1}{2\pi}\frac{d\theta(f)}{df}. \tag{3.4}$$

The phase delay of a filter represents absolute delay, and its significance is limited. The group delay, on the other hand, is used as the criterion to evaluate phase nonlinearity [11]. For linear phase responses, where $\theta(f) = -\alpha 2\pi f$ for some constant α, the group delay and phase delay are identical, and each may be seen as time delay equal to α samples.

$$\tau_p(f) \equiv -\frac{\theta(f)}{2\pi f} = -\frac{-\alpha 2\pi f}{2\pi f} = \alpha \text{ samples}, \tag{3.5}$$

and

$$\tau_g(f) \equiv -\frac{1}{2\pi}\frac{d\theta(f)}{df} = -\frac{1}{2\pi}\frac{d}{df}(-\alpha 2\pi f) = \alpha \text{ samples}. \tag{3.6}$$

If the phase response is nonlinear, then the relative phases of the sinusoidal signal components are generally altered by the filter. A linear phase variation over a band of frequencies implies a constant group delay, and no phase distortion in that band. In order to preserve the integrity of the pulse through a system, the group delay of the system must be constant.

We can formulate the compensation for the dispersion of the ionosphere as a group delay equalization problem. If a system has a non-constant group delay, we can add another filter to the processing chain to effectively cancel out the group delay of the dispersive system. We determined from testing procedures that the compression of the filter was directly dependent on the group delay match. The first part of the testing procedure consisted of generating a dispersed pulse as discussed in Chapter 1. Then, the dispersed pulse was run through the compression filter that will be described in the next sections. Invariably, the better the group delay match, the more effective the compression. This is important because we no longer need to demonstrate the compression of the filter on the dispersed waveform. If we can demonstrate that the group delay is matched more accurately, we can be confident that this results in better compression.

From the ionosphere model, we have an analytical form for the transfer function of the ionosphere. It is wholly characterized by the phase response, which is

$$\theta_{iono}(f) = 2\pi bTEC / f \,. \tag{3.7}$$

Using (2.18), the matched filter frequency response is simply given by the complex conjugate of the ionosphere frequency response. Again, the frequency response is wholly characterized by the phase response, given as

$$\theta_{matched}(f) = -2\pi bTEC / f \,. \tag{3.8}$$

Then, the group delay for the ionosphere system is

$$\tau_{iono}(f) = -\frac{1}{2\pi}\frac{d}{df}\left(2\pi bTEC / f\right) = bTEC / f^2, \tag{3.9}$$

and the group delay for the matched filter is

$$\tau_{matched}(f) = -\frac{1}{2\pi}\frac{d}{df}\left(-2\pi bTEC / f\right) = -bTEC / f^2. \tag{3.10}$$

We can see the group delay for the matched filter is negative for all frequencies. If the filter is to have a positive group delay waveform, the entire group delay waveform can be shifted up by a constant. This results in effective compression, and simply delays the output by the shifting constant. The delay is not detrimental to the solution in any manner.

We consider the group delay of the ionosphere over the passband of interest, which is 51.15 MHz to 76.725 MHz. This frequency range corresponds to the digital spectrum as shown in Figure 3.2.

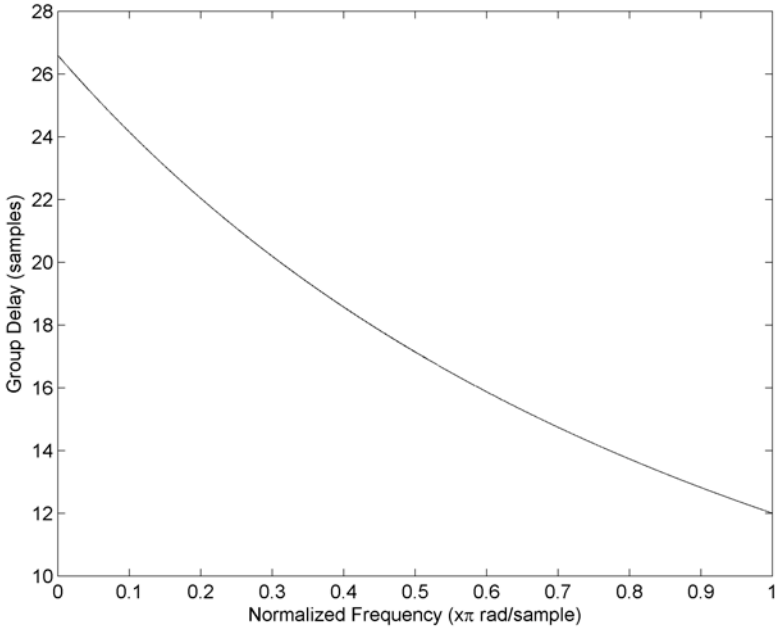


Figure 3.2 *Group delay of ionosphere.*

As mentioned, the bandwidth edges were chosen so that the sampling frequency will be a convenient fraction of the high precision GPS clock that is already available on the satellite. We must keep in mind, however, that the sampling process replicates the waveform so that the band of interest lies between 0 to 25.575 MHz. This fact simply causes a shift in the group delay expression for the system, so that it becomes

$$\tau_{iono}(f) = \frac{bTEC}{(f + f_s)^2} \tag{3.11}$$

over the frequency range $0 \le f < f_s / 2$. This can be normalized by the sampling frequency, which yields a group delay in samples:

$$\overline{\tau}_{iono}(\nu) = \frac{\overline{b}TEC}{(\nu + 1)^2}, \tag{3.12}$$
$$\text{where } \overline{b} = b / f_{s,}\ \nu = f / f_s.$$

Now, we have established two important design parameters. We have derived an expression for the ionosphere group delay, and also claimed that we need to create a filter with the same group delay, only negative. Shifting up the group delay by a constant preserves the shape of the function while ensuring the group delay is always positive. Thus, we begin to explore how we can create a filter with a prescribed group delay.

3.2 Bilinear Transform

Often, the bilinear transform is used to convert an analog transfer function into a digital transfer function [14]. To derive the digital transfer function from the analog transfer function of the system, the substitution $s = j2\pi f$ was made in the expression for the dispersive portion of the transfer function, creating:

$$H_a(f) = e^{j2\pi bTEC/f}. \tag{3.13}$$

Rearranging the substitution and solving for f, we see that $f = -js/2\pi$, so that:

$$H_a(s) = e^{-4\pi^2 bTEC/s}. \tag{3.14}$$

To continue the transformation of the continuous transfer function into the discrete-time domain, the bilinear transform will be employed. First, we substitute for s:

$$H_d(z) = H_a(s)\Big|_{s=\frac{2}{T}\frac{z-1}{z+1}} \tag{3.15}$$

$$\begin{aligned} H_d(z) &= e^{-4\pi^2 bTEC/(2(z-1)/T(z+1))} \\ &= e^{-2T\pi^2 bTEC(z+1)/(z-1)}. \end{aligned} \tag{3.16}$$

Now, for ease of notation, let $\gamma = -2T\pi^2 bTEC$, and observe that the Taylor series expansion of an exponential is given by

$$e^{x} = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^2}{3!} + \cdots \tag{3.17}$$

$$e^{\gamma(z+1)/(z-1)} \cong 1 + \frac{\gamma(z+1)}{(z-1)} + \frac{\gamma^2(z+1)^2}{2(z-1)^2} + \frac{\gamma^3(z+1)^3}{6(z-1)^3}. \tag{3.18}$$

We chose to use three terms as this captures the majority of the approximation. We multiply each term as appropriate to gain a common denominator:

$$H_d(z) = \frac{6(z-1)^3}{6(z-1)^3} + \frac{\gamma(z+1)6(z-1)^2}{(z-1)6(z-1)^2} + \frac{\gamma^2(z+1)^2 3(z-1)}{2(z-1)^2 3(z-1)} + \frac{\gamma^3(z+1)^3}{6(z-1)^3}. \tag{3.19}$$

Simplifying and gaining a common denominator, we obtain the ratio of polynomials:

$$\begin{aligned} H_d(z) &= \frac{\begin{pmatrix} z^3(6+6\gamma+3\gamma^2+\gamma^3) + z^2(-18-6\gamma+3\gamma^2+3\gamma^3) \\ +z(18-6\gamma-3\gamma^2+3\gamma^3) + (-6+6\gamma-3\gamma^2+\gamma^3) \end{pmatrix}}{6z^3 - 18z^2 + 18z - 6} \\ &= \frac{\begin{pmatrix} -(\gamma^3/6+\gamma^2/2+\gamma+1) - (\gamma^3/2+\gamma^2/2-\gamma-3)z^{-1} \\ -(\gamma^3/2-\gamma^2/2-\gamma+3)z^{-2} - (\gamma^3/6-\gamma^2/2+\gamma-1)z^{-3} \end{pmatrix}}{1-3z^{-1}+3z^{-2}-z^{-3}}. \end{aligned} \tag{3.20}$$

The matched filter for the system is given by the complex conjugate of the transfer function. Thus, in an analog sense we can state:

$$H_{match}(f) = e^{-j2\pi bTEC/f}. \tag{3.21}$$

A derivation similar to the above for the system transfer function will yield the identical transfer function $H_d(z)$, except that $\gamma = 2T\pi^2 bTEC$.

This analysis yields a filter for a TEC of 1 (with T=1/51.15 MHz) with transfer function:

$$H_d(z) = \frac{-2.3415 - 6.9974z^{-1} - 6.9705z^{-2} - 2.3146z^{-3}}{1-3z^{-1}+3z^{-2}-z^{-3}}. \tag{3.22}$$

Here we need to take note of the form of the third order IIR filter we have created. The transfer function does not have the allpass characteristic that will be described in the

next section. For this reason, it does not appear the use of the bilinear transform will be appropriate for obtaining a matched filter that can be cascaded. Without the true unity gain frequency response provided by an allpass design, higher-order cascading of the filter will undoubtedly result in a divergence from the proper compression. Any perturbation that exists in a section is compounded by the fact that multiple sections are used. This fact motivates us to use an allpass filter so that higher-order numerical robustness is achieved.

3.3 Allpass Filters

An allpass filter is one that passes all frequencies, in the sense of lowpass, highpass, and band pass filtering. The amplitude response of an allpass filter is unity at each frequency, while the phase response (which determines the delay vs. frequency) is arbitrary.
The lossless property of a filter means that the filter preserves signal energy for all input signals. That is, if the input signal is x_n, and the output signal is $y_n = h_n * x_n$, then we have

$$T\sum_{n=-\infty}^{\infty}\left|y_n\right|^2 = T\sum_{n=-\infty}^{\infty}\left|x_n\right|^2. \tag{3.23}$$

A transfer function $H(z)$ is lossless if and only if, for all *f,*

$$\left|H(e^{j2\pi f})\right| = 1. \tag{3.24}$$

Essentially, the frequency response must have magnitude 1 everywhere over the unit circle in the complex z plane [6]. For an analog system, this lossless property is very important because we need to conserve as much energy as possible from the original pulse to ensure maximum compression. For our digital system, it is less critical but it is

convenient to keep the magnitude of the transfer function at unity. We can be confident any apparent rise in the magnitude of the signal is due to the proper phase matching, not due to some arbitrary magnitude gain of the compression filter.

The transfer function of every causal, lossless IIR allpass digital filter can be written as

$$H(z) = e^{j\phi} z^{-K} \frac{\overline{A}(z)}{A(z)}$$

where $K \geq 0$, $A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_N z^{-N}$, $\overline{A}(z) = \overline{a_N} + \overline{a_{N-1}} z^{-1} + \cdots + z^{-N}$. For example, if $A(z) = 1 + 1.4z^{-1} + 0.49z^{-2}$, then $\overline{A}(z) = 0.49 + 1.4z^{-1} + z^{-2}$, which is obtained by reversing the order of the coefficients and conjugating them when they are complex. This property of allpass filters adds to their appeal because it simplifies their design. We can work with the numerator coefficients, and always know that the denominator coefficients are simply the numerator coefficients reversed (when the coefficients are real).

The unity magnitude amplitude response of the allpass filter seems to make its usefulness dubious. However, it is the phase response of the allpass filter that demonstrates its utility. Since it has no effect on the magnitude response, an allpass filter can be used for phase compensation processes. In the case we are studying here, we have said the ionosphere causes a dispersion of signals that travel through it. It is clear by review of the system model in (2.18) that the ionosphere disrupts the phase characteristic of the input signal. Therefore, to adequately compensate for the effect of the ionosphere,

we need to modify the phase of the signal coming out of the ionosphere in such a manner that the phase characteristic of the original signal is restored. It appears an effective way to accomplish this phase compensation is by creating an allpass filter with the appropriate group delay.

It can be shown that the phase response of any realizable allpass filter is always negative and monotonically decreasing [7]. The group and phase delays are functions that take positive values for a casual filter. This fact allows us to think about allpass filters as systems through which signals propagate with a frequency-dependent delay, without being subject to any absorption or amplification.

The notion of an allpass filter centers around the idea of a pole-zero pair in the z-plane [5]. The pole exists at a normalized radian frequency Ω, with a radius r. A complementary zero exists at frequency Ω, but with radius $1/r$. If we consider a pole $p = re^{j\Omega}$ and zero $\xi = (1/r)e^{j\Omega}$ (in the z-plane, with radius r and $1/r$, respectively, and with normalized radian frequency Ω), they form an allpass pair

$$A(z) = \frac{z - z_{zero}}{z - z_{pole}} = \frac{z - r^{-1}e^{j\Omega}}{z - re^{j\Omega}} = -r^{-1}e^{j\Omega}\left(\frac{-re^{-j\Omega} + z^{-1}}{1 - re^{j\Omega}z^{-1}}\right). \tag{3.25}$$

The phase term $e^{j\Omega}$ from (3.25) will not contribute to the group delay. Its derivative being zero, it will act as an additive constant. Therefore, this term in the transfer function can be neglected for our purposes. The gain term r^{-1} likewise has no effect on the group delay; it, too, will be dropped from the transfer function. For our purposes, the group delay remains unchanged if we make the simplification

$$G(z) = \frac{-re^{-j\Omega} + z^{-1}}{1 - re^{j\Omega} z^{-1}}. \tag{3.26}$$

Here, the group delay is given as:

$$\tau_{all}(\upsilon) = \frac{1 - r^2}{1 + r^2 - 2r\cos(2\pi(\upsilon - \Omega))}, \tag{3.27}$$

where $\upsilon \in [-\frac{1}{2}, \frac{1}{2}]$ Hz-s [5]. Figures 3.3 to 3.8 show the group delay that such an allpass pair produces. The plots show that the group delay is symmetric around the pole frequency, with a maximum delay

$$\tau_{\max}(\upsilon) = (1 + r)/(1 - r), \tag{3.28}$$

occurring at Ω, and a minimum delay

$$\tau_{\min}(\upsilon) = (1 - r)/(1 + r), \tag{3.29}$$

occurring at Ω+0.5 Hz-s. An important property of this pole-zero pair is that the integral of its group delay around the unit circle is always 1 Hz-s, regardless of where the pole is located [5].

The group delay can be computed in MATLAB using the function `[gd]=grpdelay(b,a,n).` This function returns the n-point group delay of the filter function described by the coefficient vectors b and a, assuming the Lth-order IIR transfer function is given in direct form as:

$$H(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_L z^{-L}}{a_0 + a_1 z^{-1} + \cdots + a_L z^{-L}} \tag{3.30}$$

For an explanation of this direct-form transfer function notation, see Appendix B.

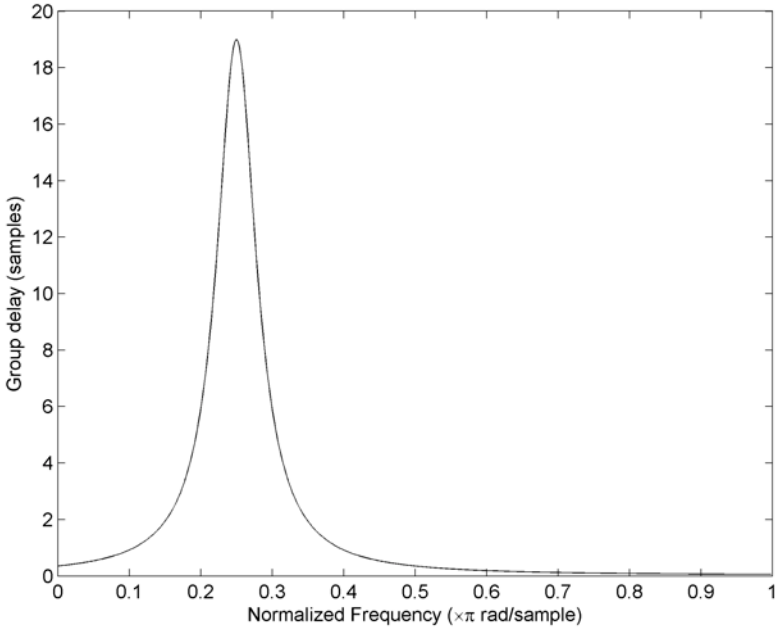


Figure 3.3 *Group Delay of Pole-Zero Pair for* r=0.9 *and* $\Omega = \pi / 4$ *rad/sample* .

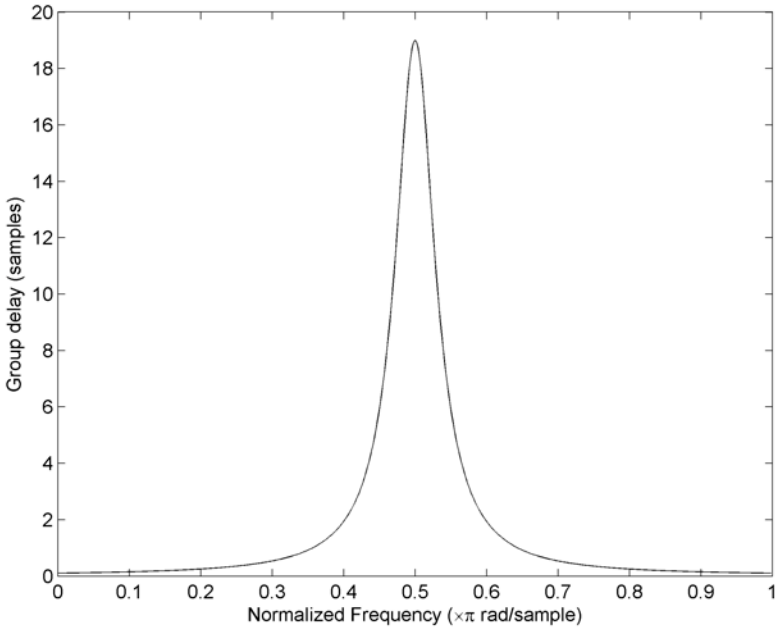


Figure 3.4 *Group Delay of Pole-Zero Pair for* r=0.9 *and* $\Omega = \pi / 2$ *rad/sample.*

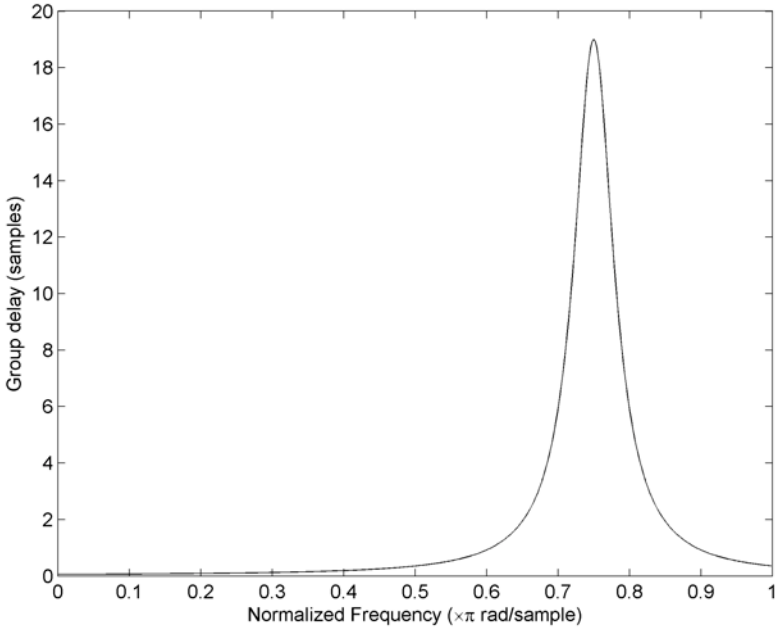


Figure 3.5 *Group Delay of Pole-Zero Pair for* r=0.9 *and* $\Omega = 3\pi / 4$ *rad/sample.*

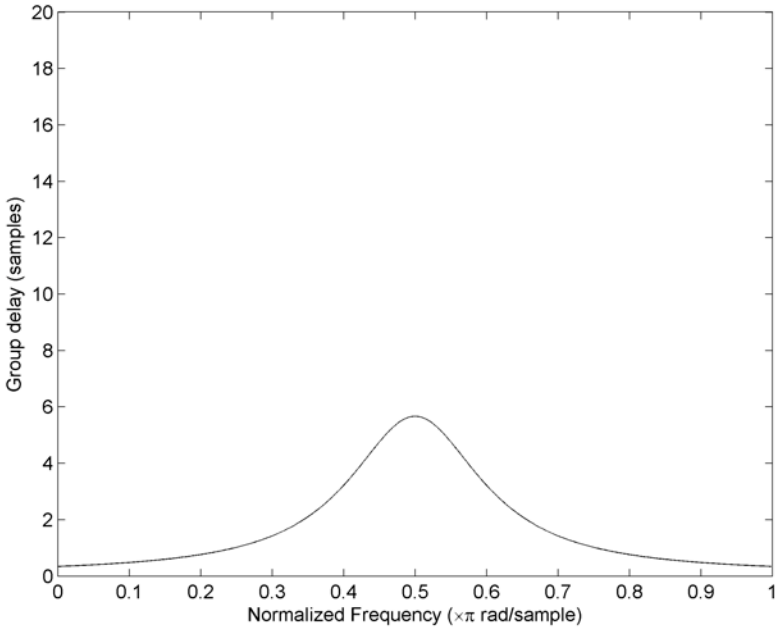


Figure 3.6 *Group Delay of Pole-Zero Pair for* r=0.7 *and* $\Omega = \pi / 2$ *rad/sample.*

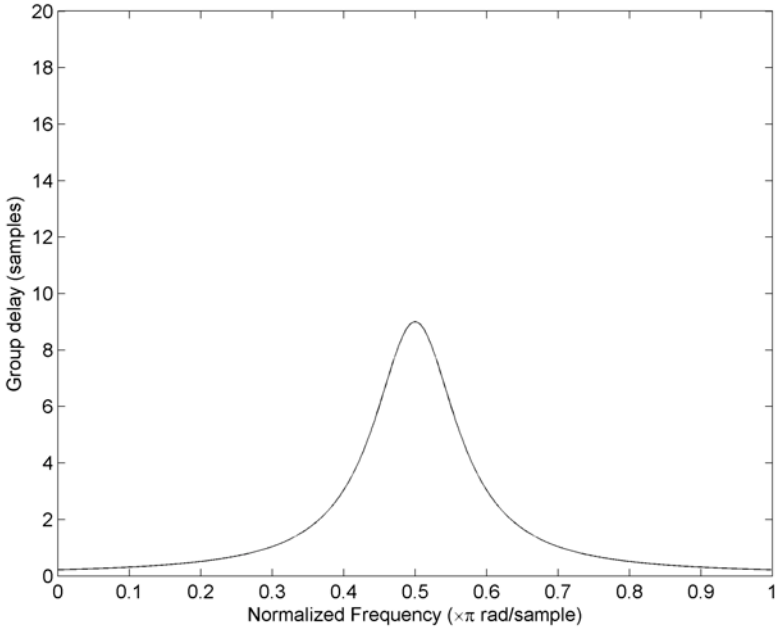


Figure 3.7 *Group Delay of Pole-Zero Pair for* r=0.8 *and* $\Omega = \pi / 2$ *rad/sample.*

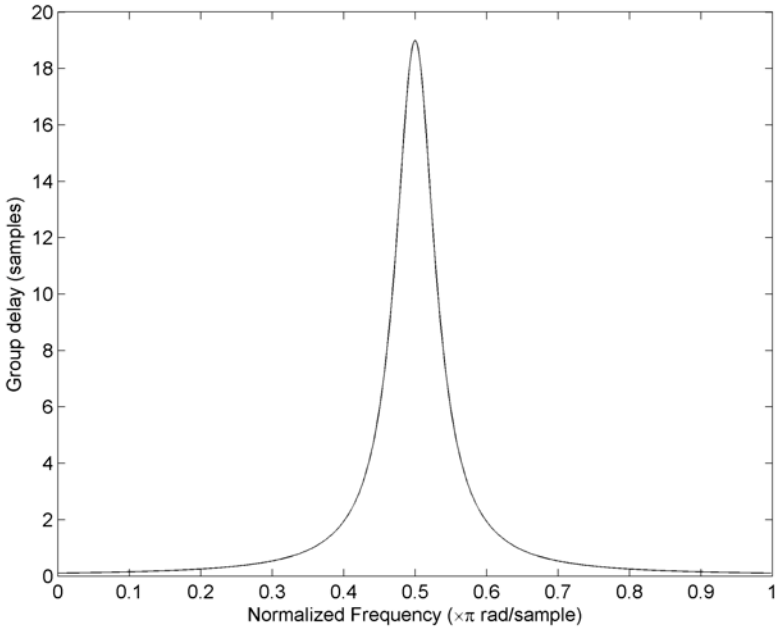


Figure 3.8 *Group Delay of Pole-Zero Pair for* r=0.9 *and* $\Omega = \pi / 2$ *rad/sample.*

We can see from these group delay figures that the center of the group delay peak coincides with the pole frequency [5]. Also, we see that the sharpness of the peak can be altered by changing the radius of the pole.

Now, we can consider the case of adding sections to the amended transfer function, so that the cascaded transfer function is of the form:

$$A_{cas}(z) = \left(\frac{-re^{-j\Omega_1} + z^{-1}}{1 - re^{j\Omega_1} z^{-1}} \right) \left(\frac{-re^{-j\Omega_2} + z^{-1}}{1 - re^{j\Omega_2} z^{-1}} \right). \tag{3.31}$$

We can see in Figure 3.9 that the resulting group delay clearly shows the contributions from each individual section pole-zero pair. This plot is indicative of the key fact that the group delay of a cascade of filter sections is equal to the sum of the individual section group delays [5].

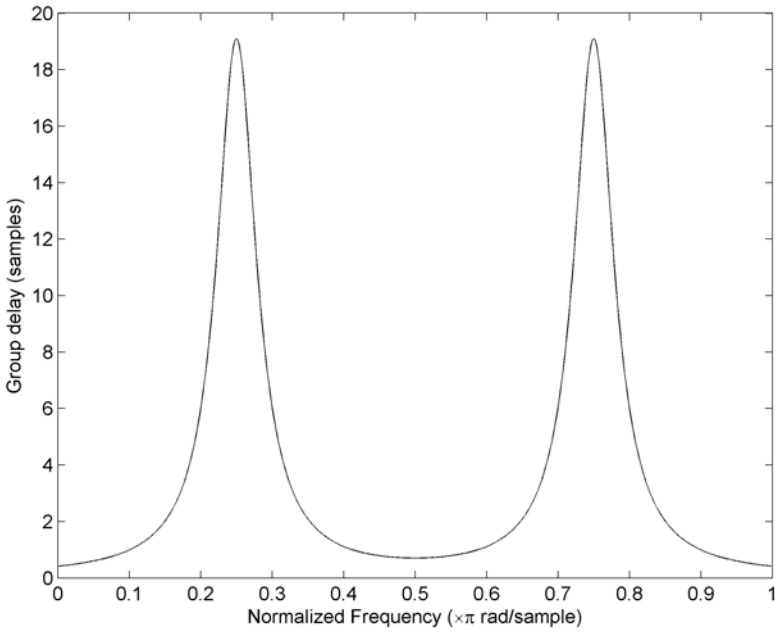


Figure 3.9 *Cascade sections for* $r = 0.9$, $\Omega_1 = \pi/4$ *rad/sample*, $\Omega_2 = 3\pi/4$ *rad/sample*.

3.4 Matched Filter Group Delay

To step through a development of the matched filter group delay, we begin with the group delay of the ionosphere system as calculated in (3.11). We will repeat the equation here for convenience:

$$\tau_{iono}(f) = \frac{bTEC}{(f + f_s)^2}. \tag{3.32}$$

To produce the group delay of the matched filter (vs. that of the system itself), we must introduce a negative sign into the group delay expression. Also, we can scale the frequency by the sampling frequency, yielding

$$\tau_{filter}(\nu) = -\frac{bTEC}{f_s^2(\nu + 1)^2}, \quad \nu = f / f_s. \tag{3.33}$$

Now, if we let $\bar{b} = bTEC / f_s$ and $\bar{\tau} = \tau_{filter} f_s$, we obtain

$$\bar{\tau}(\nu) = -\frac{\bar{b}}{(\nu + 1)^2}. \tag{3.38}$$

Since we originally defined b and f_s in terms of MHz scales, we end up with the constant $\bar{b} = 26.285$ [13]. Equation (3.38) yields the group delay function that we are ultimately trying to produce. With the proper arrangement, a cascade of such filter sections can be used to approximate a desired group delay response. Figure 3.2 shows the plot of the group delay of the ionosphere. If we can create a filter whose group delay is shaped in a similar matter, only negative (and shifted up), it will act to compress the pulse signal. This group delay plot is shown in Figure 3.10.

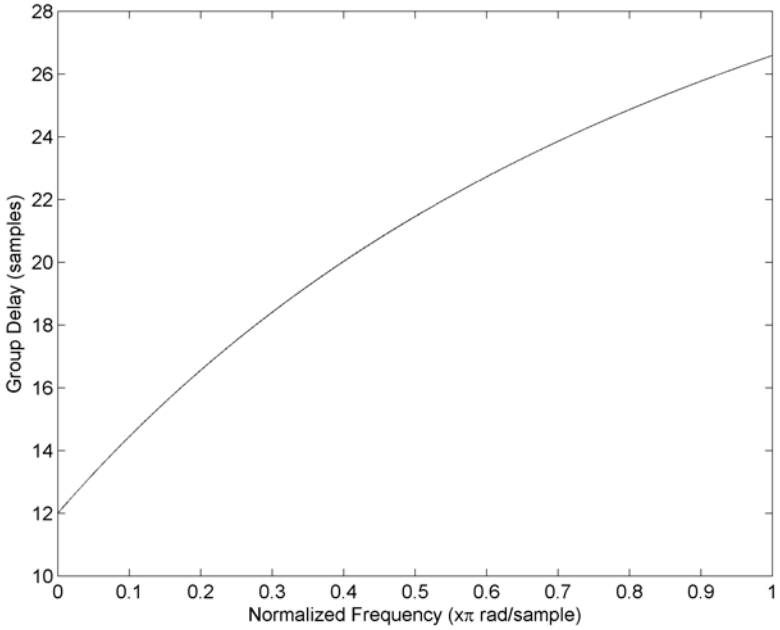


Figure 3.10 *Desired group delay of filter.*

Figure 3.10 should be compared with Figure 3.2. It is clear that the only difference here is Figure 3.10 has been negated and then shifted up; adding the two plots would result in a constant overall group delay. However, we need to realize that our filter cannot have a negative group delay and be a causal filter. We can solve this problem by adding a positive constant to the group delay plot of Figure 3.10. This will simply shift the whole waveform up by the additive constant; the cost for this is a time delay for the pulse signal. However, it is a constant time delay for all frequencies; it does not represent a phase distortion. In the next chapter, we will obtain a filter with a positive group delay having the same shape shown in Figure 3.10.

In the previous section, we introduced the allpass filter and why its properties make it an attractive choice to implement in a group delay fitting procedure. In this

section, we developed some descriptions of the group delay characteristics necessary for the anti-dispersion filter. Now, we are ready to discuss one method that accomplishes this fitting procedure in a robust manner that we emphasized was critical to a filter design that can be cascaded without cumulative divergence.

3.5 Description of the Abel-Smith Design Process

J.S. Abel and J.O. Smith, researchers in digital audio effects, described a method to extract filter coefficients for an allpass design with an arbitrary group delay [5]. One of the main ideas of the Abel-Smith procedure is the overall group delay being partitioned into subbands. Then, one allpass pair of the form in (3.26) is assigned to each subband. The more subbands the group delay is partitioned into, the better the fit to the group delay plot. Therefore, using more subbands means using more sections, increasing the order of the overall filter. The more closely we desire to match the group delay waveform, the higher the order of the resultant filter must be. As discussed in the last section, the cascaded arrangement of such filters reproduces the desired group delay characteristic because the phases (and therefore group delays) of cascaded filters are additive [7].

The design procedure is as follows:

1. Add a constant delay to the desired frequency-dependent delay so that it integrates to a desired integer value. This integer value is the order of the filter.
2. Starting at DC, divide the group delay plot into unity-area frequency subbands.
3. Fit a first-order (complex) allpass section to each subband.
4. Cascade the first-order sections to form the allpass filter.

According to the work done by Abel and Smith, we first need to set the integral of the group delay to a positive integer value. The reason for this is that the filter is

designed using sections each having a pole, and each pole contributes an area of 1 to an integration of the group delay over the unit circle [5]. That is, each pole contributes a phase accumulation of 2π radians [5]. So, we add a constant delay offset $\overline{\tau}_{off}$ to the function $\overline{\tau}(\nu)$ to obtain

$$\tau_{comp}(\nu) = \overline{\tau}(\nu) + \overline{\tau}_{off}. \tag{3.38}$$

Now, to calculate the proper offset, we need to evaluate the integral:

$$\int_0^{0.5} \tau_{comp}(\nu)\, d\nu = \int_0^{0.5} -\frac{\overline{b}}{(\nu+1)^2} + \overline{\tau}_{off}\, d\nu = N \tag{3.39}$$

$$\left(\overline{\tau}_{off}\nu + \overline{b}/(\nu+1)\right)\Big|_{\nu=0}^{0.5} = N$$

$$\overline{\tau}_{off} = 2N + 2\overline{b}/3.$$

Notice that the integration was carried out only to 0.5 Hz-s; this is because the group delay is an even function for a filter with real coefficients [5]. Now, we have formulated a new group delay function that consists of the matched filter group delay, to which we added a constant offset to ensure causality of the filter. Then, the group delay of the compensator filter is

$$\overline{\tau}_{comp}(\nu) = 2N + 2\overline{b}/3 - \overline{b}/(\nu+1)^2. \tag{3.40}$$

Since we designed the integral of the compensator group delay to be a positive integer, we are ready to divide this group delay into subbands with unity area.

Following now into step 2, for a given subband, we define the lower and upper subband edges as υ_- and υ_+, respectively. We can calculate what the proper subband edges are by setting the integral of the group delay over the subband equal to unity. Denoting the first frequency band edge as ν_1, the calculation then becomes:

$$\int_0^{\nu_1}\left(\overline{\tau}_{off}-\frac{\overline{b}}{(\nu+1)^2}\right)d\nu=1 \tag{3.41}$$

Performing the definite integration yields a quadratic expression in ν_1:

$$\overline{\tau}_{off}\nu_1^{\,2}+(\overline{\tau}_{off}-\overline{b}-1)\nu_1-1=0\,. \tag{3.42}$$

This function can be solved for ν_1 using the quadratic formula. Now, we realize that the remaining subband edges are dependent on the previous subband edge. That is, for each subband the upper frequency edge is dependent on the lower frequency edge. Thus, we can write the definite integral formula:

$$\begin{gathered}\int_{\nu_{n-1}}^{\nu_n}\left(\overline{\tau}_{off}-\frac{\overline{b}}{(\nu+1)^2}\right)d\nu=1\\ \left(\overline{\tau}_{off}\nu+\frac{\overline{b}}{(\nu+1)}\right)\Bigg|_{\nu_{n-1}}^{\nu_n}=1\\ \overline{\tau}_{off}\nu_n^{\,2}+(\overline{\tau}_{off}-\overline{\tau}_{off}\nu_{n-1}-\overline{b}\,/(\nu_{n-1}+1)-1)\nu_n\\ +\overline{b}-\overline{\tau}_{off}\nu_{n-1}-\overline{b}\,/(\nu_{n-1}+1)-1=0\end{gathered} \tag{3.43}$$

This equation is quadratic in ν_n, and using the positive root of the quadratic formula we can state:

$$\begin{aligned}\nu_n=&\frac{-(\overline{\tau}_{off}-\overline{\tau}_{off}\nu_{n-1}-\overline{b}\,/(\nu_{n-1}+1)-1)}{2\overline{\tau}_{off}}\\ &+\frac{\sqrt{(\overline{\tau}_{off}-\overline{\tau}_{off}\nu_{n-1}-\overline{b}\,/(\nu_{n-1}+1)-1)^2-4\overline{\tau}_{off}(\overline{b}-\overline{\tau}_{off}\nu_{n-1}-\overline{b}\,/(\nu_{n-1}+1)-1)}}{2\overline{\tau}_{off}}\end{aligned} \tag{3.44}$$

For the case with N=5 sections, the band edges are:

$\nu_0=0,\nu_1=0.1863,\nu_2=0.2838,\nu_3=0.3637,\nu_4=0.4347,\nu_5=0.5,$ all in Hz-s.

Now, we can move on to the third step of the Abel-Smith procedure, which involves fitting an allpass filter to each sub band. To do this, we must find the pole location for each subband. The pole-zero pair delay should approximate the desired group delay in that subband, and be small outside the subband. The pole frequency is taken to be the midpoint of the subband, that is:

$$\Omega_n = (v_{n+} + v_{n-})/2. \tag{3.45}$$

The pole radius is chosen so that the group delay at either subband edge is a fraction β of the peak group delay. The pole radius is given as [5]:

$$r(\beta) = \left(\frac{1-\beta\cos\Delta}{1-\beta}\right) - \left[\left(\frac{1-\beta\cos\Delta}{1-\beta}\right)^2 - 1\right]^{\frac{1}{2}}, \tag{3.46}$$

where $\Delta = (\upsilon_+ - \upsilon_-)/2$. In the case that the sub bands are numerous, the approximation may be made that:

$$r(\beta) \approx 1 - \Delta\sqrt{(\beta/(1-\beta)}, \quad \Delta << 1. \tag{3.47}$$

Equation (3.46) yields the radius of the pole for the section. Here, β is the fraction of the peak group delay that is present at either band edge. If β is near unity, then the group delay is smoother, since the value at each band edge is nearly equal to the peak value. If β is smaller, then the values at the band edges are likewise smaller compared to the peak value. This parameter β controls the width of the group delay peak to determine the tradeoff between smoothness of fit and ability to follow small-bandwidth features in the delay curve. A review of Figures 3.6 through 3.8 will make this idea more

clear. Now that we have equation (3.46) which yields the section pole radius, the pole frequency is taken to be the center of the subband,

$$\Omega = \left(\upsilon_+ + \upsilon_-\right)/2. \tag{3.48}$$

We use equations (3.46) and (3.48) once for each subband, yielding a pole with radius r at frequency Ω, and a corresponding zero at radius $1/r$, also at frequency Ω.

The Abel-Smith algorithm creates *N* complex numbers, corresponding to poles in *N* first order all-pass sections. Each section is paired up with its complex conjugate section, which results in the formation of a bi-quadratic section with real coefficients. The derivation of the bi-quadratic begins with assigning the pole for section n at $s_n = r_n e^{j\Omega_n}$, and the corresponding complex conjugate pole at $s_n^* = r_n e^{-j\Omega_n}$. Placing these values into equation (3.31), we develop an expression for the bi-quadratic section as

$$\begin{aligned} H_n(z) &= \left(\frac{-s_n^* + z^{-1}}{1 - s_n z^{-1}}\right)\left(\frac{-s_n + z^{-1}}{1 - s_n^* z^{-1}}\right) \\ &= \frac{s_n s_n^* - s_n^* z^{-1} - s_n z^{-1} + z^{-2}}{1 - s_n^* z^{-1} - s_n z^{-1} + s_n s_n^* z^{-2}} \\ &= \frac{s_n s_n^* - z^{-1}(s_n^* + s_n) + z^{-2}}{1 + z^{-1}(s_n^* + s_n) + s_n s_n^* z^{-2}} \\ &= \frac{r_n^2 - 2r_n \cos(2\pi\Omega_n) z^{-1} + z^{-2}}{1 - 2r_n \cos(2\pi\Omega_n) z^{-1} + r_n^2 z^{-2}}, \end{aligned} \tag{3.49}$$

where we used

$$\begin{aligned} s_n s_n^* &= \left(r_n e^{j\Omega_n}\right)\left(r_n e^{-j\Omega_n}\right) = r_n^2 \\ s_n^* + s_n &= r_n\left(e^{j\Omega_n} + e^{-j\Omega_n}\right) = 2r_n \cos(2\pi\Omega_n). \end{aligned}$$

This second order transfer function exhibits the allpass characteristic of the denominator weight vector being a reversed numerator weight vector.

The group delay of a single pole allpass filter was given in (3.27), and is repeated here:

$$\tau_{all}(\nu) = \frac{1-r^2}{1+r^2-2r\cos(2\pi(\nu-\Omega))}. \tag{3.50}$$

The group delay of the bi-quadratic section will be the sum of (3.50) with the group delay of the conjugate of (3.50), as shown in (3.51). Thus, we can state:

$$\tau_{bi}(\nu) = \left(\frac{1-r^2}{1+r^2-2r\cos(2\pi(\nu-\Omega))}\right) + \left(\frac{1-r^2}{1+r^2-2r\cos(2\pi(\nu+\Omega))}\right) \tag{3.51}$$

The bi-quadratic section group delay in (3.51) is important because it is the building block of the cascaded filter.

A biquad filter is defined by the following flowgraph or difference equation shown in Figure 3.11.

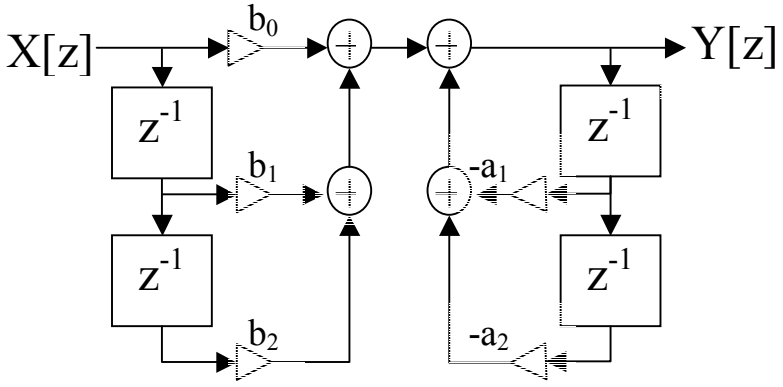


Figure 3.11 *Biquad filter flowgraph.*

Notice that both the numerator and denominator of the transfer function are quadratic in z, hence the name of the filter (short for "bi-quadratic"). We can derive the difference equation corresponding to the flowgraph of Figure 3.11:

$$\begin{aligned} y[n] &= b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] - a_1 y[n-1] - a_2 y[n-2] \\ Y(z) &= b_0 X(z) + b_1 X(z) z^{-1} + b_2 X(z) z^{-2} - a_1 Y(z) z^{-1} - a_2 Y(z) z^{-2} \\ Y(z)(1 + a_1 z^{-1} + a_2 z^{-2}) &= X(z)(b_0 + b_1 z^{-1} + b_2 z^{-2}) \\ \frac{Y(z)}{X(z)} &= \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \end{aligned} \tag{3.52}$$

For our development, we can see that (3.52) is in the same form as (3.49), and reading the coefficients directly we have:

$$\begin{aligned} b_0 &= r_n^2 \\ b_1 &= -2r_n \cos(2\pi\Omega_n) \\ b_2 &= 1 \\ a_1 &= -2r_n \cos(2\pi\Omega_n) \\ a_2 &= r_n^2. \end{aligned} \tag{3.53}$$

Now, we have (3.46), (3.47), and (3.48) to calculate the pole radius and frequency for each section. Using Equation (3.53) we can calculate the coefficients for each biquad section. Since every biquad filter is 2nd order, when we cascade *N* sections together, the overall filter is of order 2*N*. The MATLAB code that creates the filter sections as well as the overall filter coefficients is included in Appendix A.

3.6 Optimization of the Allpass Filter

In the previous section, we mentioned the parameter β as controlling the width of the allpass filter group delay plots. This behavior is shown in Figures 3.6-3.8. Abel and Smith referred to this value as a user-supplied parameter. In their implementation, the β

value is held constant for every subband. At the outset, there is no way for the user to know what value of β will yield the best approximation of the desired group delay. In practice, the user chooses a value of β by a "cut-and-try" method. In our implementation of the Abel-Smith approach, the β value resulting in the lowest group delay error was found empirically. The code used to create the Abel-Smith filter is listed in Appendix E. The β value could be computed as a distinct value for each biquad section. Abel and Smith actually suggest this themselves to create an improved filter [5]. Using a different β value for each section would undoubtedly create a more closely matched group delay plot. However, again we have no procedure to follow to obtain the best β value for each of the sections. These facts motivate the shift of our focus to an optimization paradigm.

Optimization can be described as the minimization or maximization of a function subject to constraints on its variables. This function *f(x)* is termed the objective function, and it is what is actually minimized or maximized. The objective function is a function of the variables (also called unknowns or the parameters) *x*. The *x* values are those quantities or factors that can be adjusted to produce a minimum or maximum objective function value. There are also constraints associated with the whole process. They place bounds on the values the variables may take on, and are denoted $c_i(x)$. The constraints stem from the fact that the solution to every practical problem has limits placed on it. The limits may be man-made (cost), or may be of an inherent physical nature (stability, causality, conservation of energy). The term "optimization", then, means the implementation of an algorithm that finds the values of the variables which result in the

minimum or maximum objective function value. The values of the variables that are produced, while respecting the constraints, are termed the "optimal solution".

At this point we have specified the basic parameters that optimization routines are concerned with. Now we are ready to discuss how to actually carry out the optimization, and we begin the next section with a brief overview of Newton's method. This method is a common element in two optimization algorithms we will mention.

3.6.1 Newton's Method

Newton's method can be used to find local maxima and minima of functions. The method is based on noticing that if a real number x^* is a stationary point (a point where the gradient is zero; equivalently, where the function ceases to increase or decrease) of a function $f(x)$, then x^* is a root of the derivative of $f(x)$. Therefore, an application of Newton's method to $f(x)$ can solve for x^*. We consider a variable vector x and let $\delta = [\delta_1 \ \delta_2 \ \cdots \ \delta_K]^T$ be a change in x. The Taylor series at point $x + \delta$ is given by:

$$f(x+\delta) = f(x) + \delta \nabla f(x) + \frac{1}{2} \delta H \delta^T, \tag{3.54}$$

where the gradient vector is defined as

$$\nabla f(x) = \left[\frac{\partial f(x)}{dx_1} \quad \frac{\partial f(x)}{dx_2} \quad \cdots \quad \frac{\partial f(x)}{dx_K} \right]^T, \tag{3.55}$$

and the Hessian matrix is defined as

$$H = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_K} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_K} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_K \partial x_1} & \frac{\partial^2 f(x)}{\partial x_K \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_K^2} \end{bmatrix}. \tag{3.56}$$

We will discuss two strategies associated with Newton's method: the line search and the trust region.

3.6.2 Line Search

In the line search strategy, the algorithm chooses a direction p_k and searches along this direction from the current position x_k for a new position with a lower objective function value. In a vector sense, we need both a direction and a magnitude to the new point. The distance to move along p_k can be found by approximately solving:

$$\min f(x_k + \alpha p_k), \quad \alpha > 0. \tag{3.57}$$

The line search algorithm generates a number of trial steps until it finds one that approximates (3.57). Then, the process is repeated at the new point. The steepest decent direction $-\nabla f_k$ is the most obvious choice for the search direction. Or, we can rearrange (3.54) to obtain the Newton direction, $-H^{-1}\nabla f(x_k)$. The transition to the next position is then given by:

$$x_{k+1} = x_k - H^{-1}\nabla f(x_k), \quad k \geq 0. \tag{3.58}$$

A geometric interpretation of Newton's method is that for every iteration of (3.58), $f(x)$ is approximated by a quadratic function around x_k, and then a step is taken to the minimum of that quadratic function.

The method we chose to implement the line search algorithm is called the least-pth method. The algorithm minimizes:

$$\Psi_k(x) = E(x)\left\{\sum_{i=1}^{K}\left[\frac{|e_i(x)|}{E(x)}\right]^p\right\}^{1/p}, \quad E(x) = \max|e_i(x)| \tag{3.59}$$

The error values $e_i(x)$ are defined as the differences between the actual group delay and the ideal group delay:

$$e_i(x) = M_i(x) - M_{ideal,i}(x), \quad 1 \le i \le K. \tag{3.60}$$

There is a listing of pseudo-code to perform the least-pth method [10] in Appendix C. Besides the line search, another newer method in use is the trust region method.

3.6.3 Trust Region

In the trust region strategy, the information gathered about the objective function *f(x)* is used in the construction of a model function m_k. Consider a case where the objective function is a curve, and currently we are considering a point, x_k, on that curve (where the subscript denotes the iteration index). The behavior of m_k near the current point x_k is similar to that of $f(x_k)$. Since m_k is only a good approximation of f near x_k, the search for a minimizer of m_k is restricted to a trust region, Δ_k, around x_k. A

search is conducted for the minimum of m_k inside the trust region by evaluating a candidate step. The candidate step p is found by approximately solving:

$$\min m_k(x_k + p), \tag{3.61}$$

where $x_k + p$ is inside the trust region. If $f(x_{k+1}) < f(x_k)$ by a sufficient amount, then we have a success: m_k is a good local approximate of *f(x)*. Now, we move to position x_{k+1} and iterate (*k=k+1*). We gain confidence and increase the trust region radius Δ_k. Next, a new quadratic model m_k is constructed around the new x_k. If we have a partial success, where the reduction is not as big as predicted by m_k, then we still move x_k, but leave Δ_k unchanged. If the model fails so that $f(x_{k+1}) > f(x_k)$, it means m_k is not a good enough approximate of *f(x)*. We lose confidence in the model, the radius Δ_k is reduced, and we do not change the current position x_k.

The model m_k is usually defined to be a quadratic function of the form:

$$m_k(x_k + p) = f_k + p^T \nabla f_k, \tag{3.62}$$

where f_k, and ∇f_k are a scalar and vector, respectively. The solution to this problem is:

$$p_k = -\frac{\Delta_k \nabla f_k}{\|\nabla f_k\|}, \tag{3.63}$$

where $\|.\|$ denotes the L2-norm.

We have discussed some general algorithms that can be used in an optimization effort. We are now ready to speak about our problem in the language of optimization.

3.6.4 Formulation in Optimization Terms

In our approach, the variables being used are the radius and frequency of the pole for each filter section. As discussed in the section describing allpass filters, the radius of the zero for each section is the inverse of the radius of the pole for that section. Also, the frequency of the zero is the same as that of the pole. Therefore, the variables need only include the pole radii and frequencies, from which the zero locations can be calculated. Since the number of sections N in the optimal filter can be varied (ultimately varying the order of the overall filter), the number of variables also varies. This results in the variable vector x being of a length equal to twice the number of sections N. Thus, we can form x as

$$x = \left[r_1, r_2, \ldots, r_N, \Omega_1, \Omega_2, \cdots, \Omega_N \right]^T . \tag{3.64}$$

The constraints on the variables are based on the stability criterion of the poles being within the unit circle. This means the pole radii are constrained to:

$$0 < r_1, r_2, \ldots, r_N < 1, \tag{3.65}$$

and the normalized radian frequencies are constrained so that:

$$0 < \Omega_1, \Omega_2, \ldots, \Omega_N < 2\pi . \tag{3.66}$$

The ideal group delay τ_{ideal} was shown in Figure 3.10; it is the group delay we desire the optimal filter to have. The optimal group delay $\tau_{opt}(x)$ is the group delay of the overall filter, which includes all the cascaded sections. Equation (3.50) lists the group delay of a single biquad section; since N biquads are cascaded, we can write the group delay of the optimal filter as:

$$\tau_{opt}(x) = \sum_{n=1}^{N}\left(\frac{1-r_n^{\,2}}{1+r_n^{\,2}-2r_n\cos(2\pi(\nu-\Omega_n))}\right)+\left(\frac{1-r_n^{\,2}}{1+r_n^{\,2}-2r_n\cos(2\pi(\nu+\Omega_n))}\right). \quad (3.67)$$

We are defining the objective function as the difference between the desired group delay response, and the group delay of the filter the optimization algorithm produces. Therefore, the objective function $f(x)$ is defined in terms of the ideal group delay sequence τ_{ideal} and optimal group delay sequence $\tau_{opt}(x)$:

$$f_{err}(x) = \sum_{i=1}^{L}(\tau_{ideal,i} - \tau_{opt,i}(x))^2 \quad (3.68)$$

The index *i* is meant to indicate that the group delays are vectors, calculated at discrete frequencies, each of length *L*. *L* is simply the number of points in the sequences. The optimization procedure finds the values in the *x* vector that produce a filter whose group delay $\tau_{opt}(x)$ is the optimal approximation of τ_{ideal}, thus minimizing $f_{err}(x)$. The summation in equation 3.68 makes the objective function a scalar value. The group delay functions are both vectors since they are functions of the frequency index.

The MATLAB code *iirgrpdelay* is the implementation of the method that we used to apply the least p-th method. The code used is available in Appendix A.

We applied the interior-reflective Newton's method to form the final solution vector. We chose to implement this trust region algorithm with a routine designed to solve non-linear least squares problems. Pseudo-code for this method is available in Appendix D. While it is calculating the solution vector, the algorithm will not allow the

elements in the vector to be outside the lower and upper bounds. We began the optimization initializing with pole locations that were computed by the Abel-Smith algorithm. This automatically set the order of the optimized filter to the order of the Abel-Smith initializing filter. However, to verify robust behavior we tested the algorithm using a randomly generated number set to represent the pole locations. The results of this testing are contained in the next chapter.

Chapter 4

Results

Now that we have put forth the effort to create an optimization routine, we can take a look at some results and see if we are being rewarded.

4.1 Group Delay Fit

The resultant fit of the Abel-Smith filter is shown in Figure 4.1. The value of β was chosen empirically to minimize the MSE between the group delay of the filter and the ideal group delay.

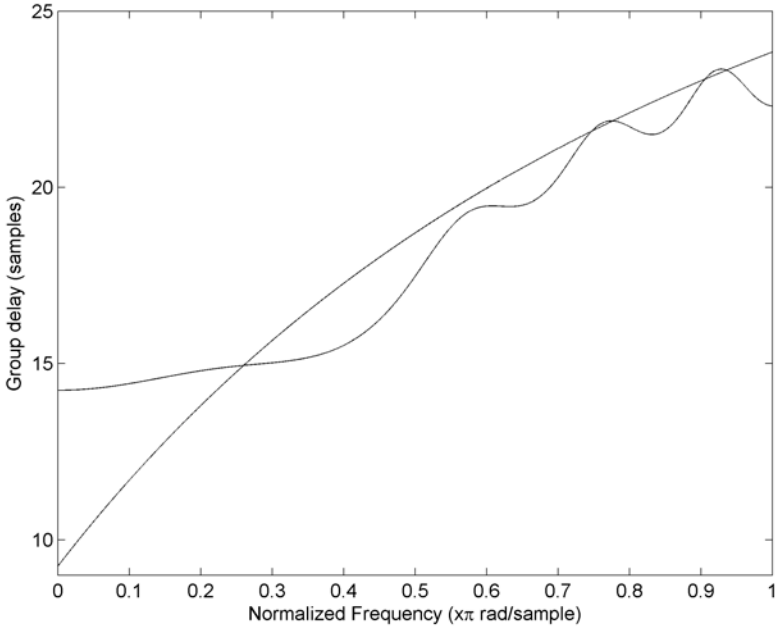


Figure 4.1 *Abel-Smith 8th order filter group delay fit,* β=0.57.

We can see that the fit provided by the Abel-Smith filter is not very accurate with this low-order filter. A higher order filter is shown in Figure 4.2.

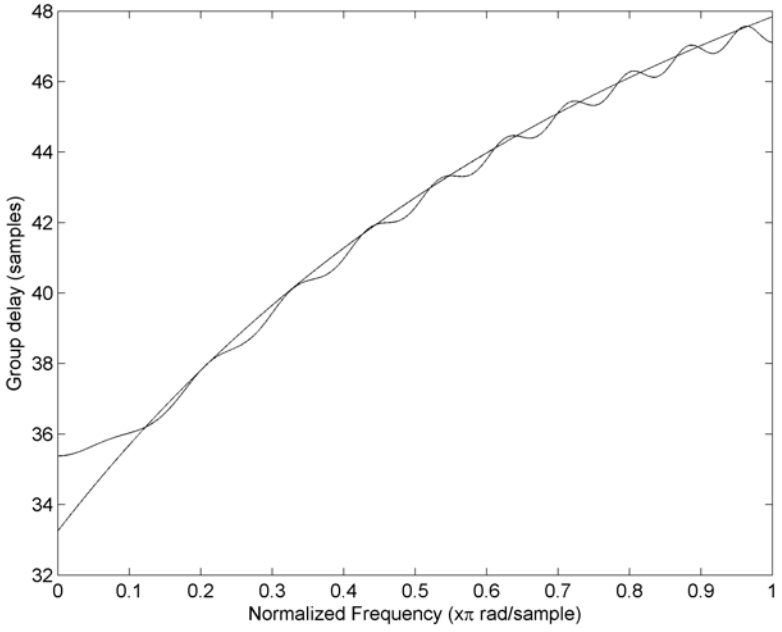


Figure 4.2 *Abel-Smith* 20^{th} *order filter group delay fit,* β=0.74.

As we would expect, the fit becomes closer as the order of the filter is increased. To obtain close fits that Abel-Smith demonstrate in [5], the orders of the filter were between 40 and 128. If we were allowed the use of a very high order filter, the Abel-Smith method would work nicely. However, the reality of the situation is that the filter order is constrained to be relatively low. Radiation hardened FPGAs will be used for the system, and their computational ability is a limiting factor for the filter.

As mentioned, we desired the filter design to exhibit lossless behavior so we do not waste any available signal energy. A plot of the magnitude response in Figure 4.3 shows that the filter gain is essentially unity.

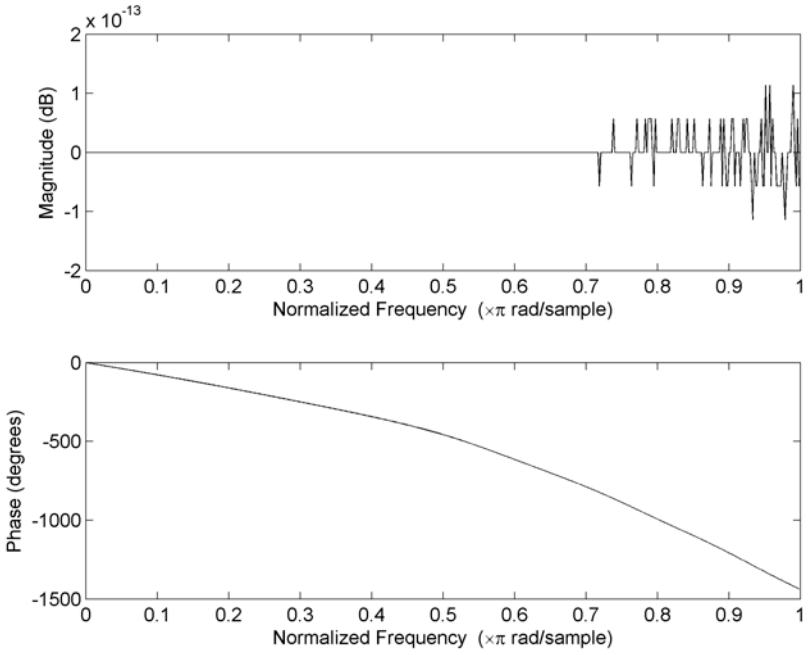


Figure 4.3 *Abel-Smith 8th order filter frequency response.*

Abel and Smith pointed out that the phase accumulation is -2π per pole (-360 degrees). So, for the 4 poles in the upper half of the plane we expect the cumulative phase to be -1440 degrees. We see that the ultimate value in the phase response of Figure 4.3 is indeed -1440 degrees. So, the filter is showing allpass behavior as we expect.

There is another very important filter quality that we need to evaluate: filter stability. We will check the stability of the systems by viewing the pole-zero plots they produce. Figure 4.4 shows the pole-zero plot for the Abel-Smith filter for an 8th order case.

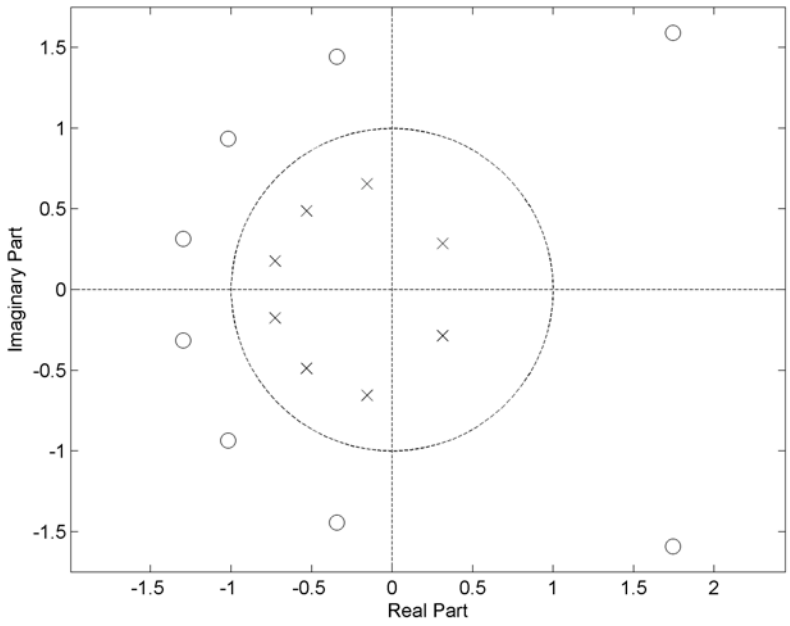


Figure 4.4 *Abel-Smith 8th order filter pole-zero map.*

In Figure 4.4 we can clearly see the complimentary zeros for each pole. The zeros are at the same frequencies as the poles, but with radii equal to the inverse of the pole radii. Also, we can see the biquad characteristic that the poles and zeros come in complex conjugate pairs. Table 4.1 lists the pole radii for the filter, and indicates whether they meet the stability criterion. The criterion is that the pole radii are less than unity.

Pole Number	Pole Radius	Meets Stability Criterion?
1	0.4237	Yes
2	0.4237	Yes
3	0.6737	Yes
4	0.6737	Yes
5	0.7224	Yes
6	0.7224	Yes
7	0.7488	Yes
8	0.7488	Yes

Table 4.1 *Abel-Smith pole radius listing and stability conformity*

We can see from Table 4.1 that the Abel-Smith filter is stable, since all of the poles meet the stability criterion.

Now, we will look at the behavior that the optimized filter demonstrates. First, we will look at the group delay fit in Figure 4.5. We can see the fit is remarkably better for the unconstrained optimized version of the filter. With the same filter order, the MSE is hundreds of times smaller for the unconstrained optimized version than for the Abel-Smith design.

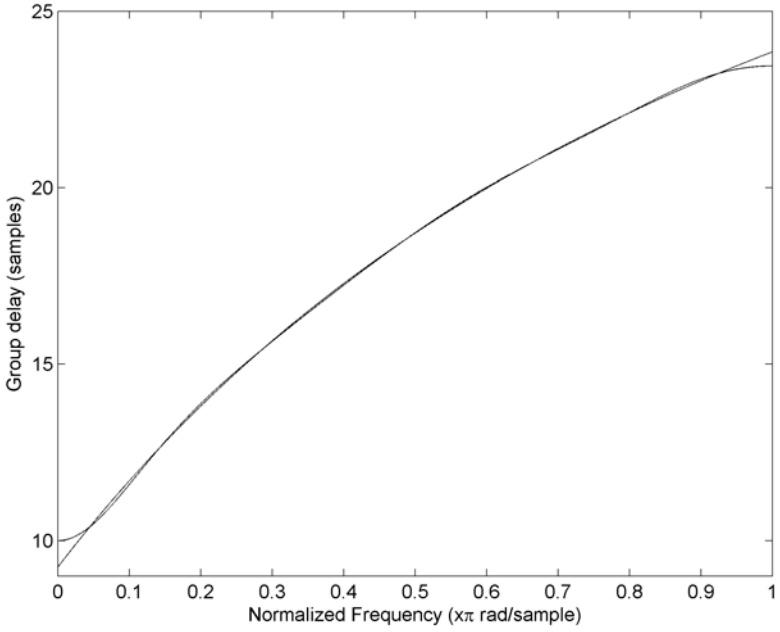


Figure 4.5 *Unconstrained optimized 8th order filter group delay fit.*

We can check if the optimized filter is exhibiting allpass behavior by observing the frequency response. The response for an 8th order optimized filter is shown in Figure 4.6.

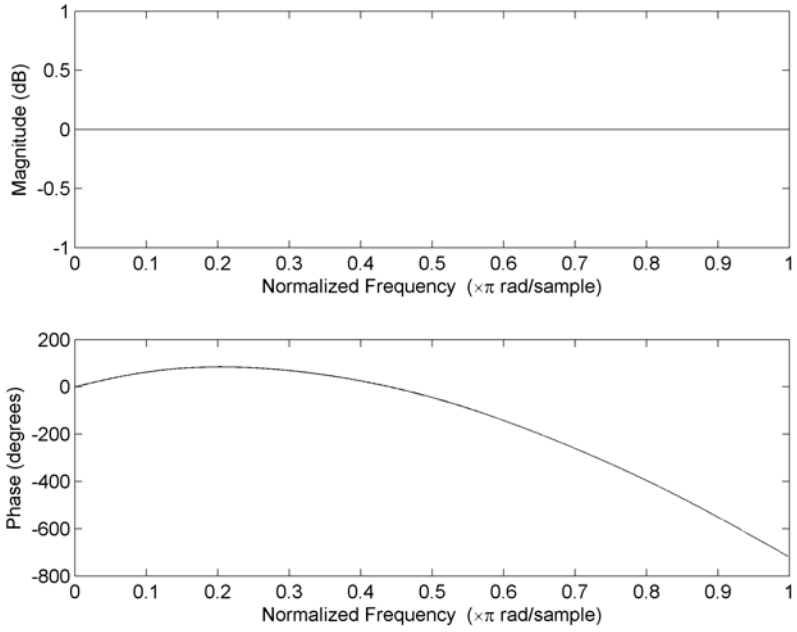


Figure 4.6 *Unconstrained optimized 8th order frequency response.*

In Figure 4.6, the magnitude response is exhibiting allpass behavior. The phase response, however, is not exhibiting allpass characteristics. We stated previously that the phase response of every realizable allpass filter must be always negative and monotonically decreasing [7]. Clearly, the phase response in Figure 4.6 is exhibiting neither of these characteristics. The pole-zero plot in Figure 4.7 will give us more insight into what is occurring with the unconstrained optimized filter.

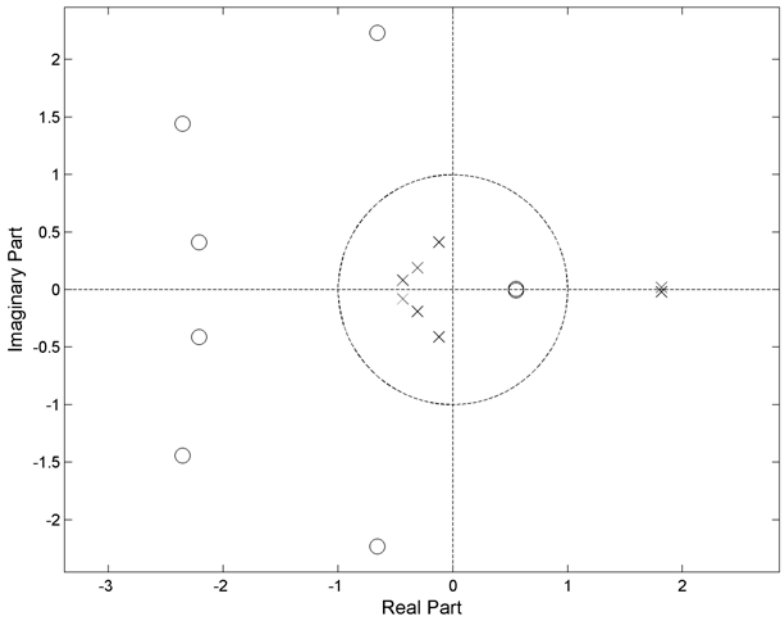


Figure 4.7 *Unconstrained optimized 8th order pole-zero map.*

Pole Number	Pole Radius	Meets Stability Criterion?
1	0.3623	Yes
2	0.3623	Yes
3	0.4298	Yes
4	0.4298	Yes
5	0.4450	Yes
6	0.4450	Yes
7	1.8162	No
8	1.8162	No

Table 4.2 *Unconstrained optimized 8th order pole radius listing and stability conformity*

The last pair of lines in Table 4.2 are the source of the problem with the filter. It fails the stability criterion with poles that lie outside the unit-circle on the z-plane. Clearly, we can see these errant poles in the plot of Figure 4.8. In spite of the excellent group delay fit the filter produces, it will be of no value to us if it is not a stable design. The solution to this problem was described in the optimization section. The optimization routine was re-written to enforce a constraint of the pole radii present in the final filter design. We will now discuss the results that this constrained optimization produced, and disregard the unstable filter that the unconstrained approach generated.

4.2 Performance Metrics

The metric used to gauge the performance of the filters was chosen to be the squared error and the mean squared error. The squared error (SE) is calculated by squaring the point by point difference between the ideal group delay and the group delay of the filter. Since the operation is point by point, the squared error remains a vector:

$$SE_i = (\tau_{ideal,i} - \tau_{act,i})^2.$$

The mean squared error (MSE) is simply the mean of the squared error, and hence is a scalar:

$$MSE = \frac{1}{L}\sum_{i=1}^{L}(\tau_{ideal,i} - \tau_{act,i})^2.$$

Various filter orders yielded MSE values listed in Table 4.3. We can see that the optimized filters create significantly lower MSE values.

Filter Type	Order	MSE
Abel-Smith	8	2.6079
Line Search	8	0.7291
Trust Region	8	0.5443
Abel-Smith	10	1.4939
Line Search	10	0.4415
Trust Region	10	0.1953
Abel-Smith	14	0.5782
Line Search	14	0.1220
Trust Region	14	0.0284
Abel-Smith	20	0.2098
Line Search	20	0.0242
Trust Region	20	0.0044

Table 4.3 *Comparison of MSE measured for each filter*

Plots showing the ideal group delay vs. the group delay obtained for a 10^{th} order filter of each variety are shown in Figure 4.8-4.10. We can see that the group delay is markedly better for the optimized filters. The lack of control over β is apparent in the Abel-Smith plot, as it shows substantially more error over the whole range. The plot was made with a β value that was found empirically to minimize the MSE.

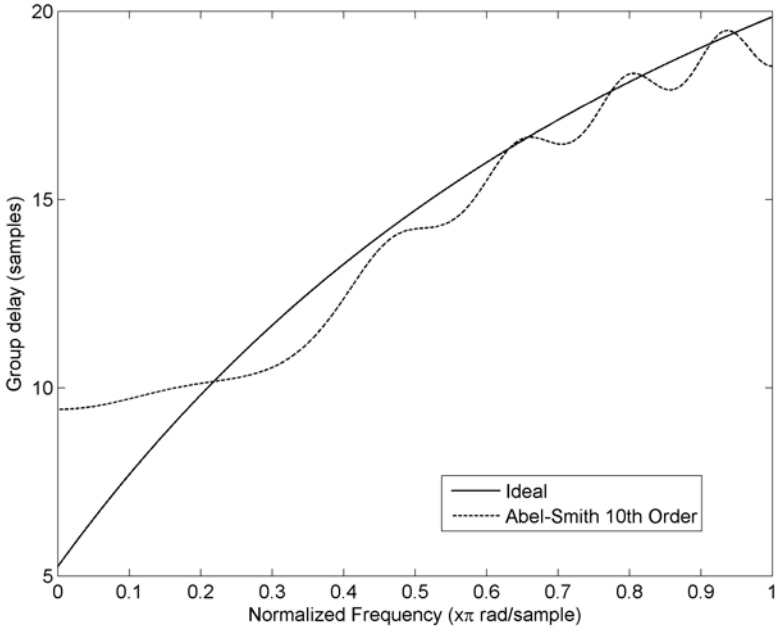


Figure 4.8 *Abel-Smith 10th order filter group delay.*

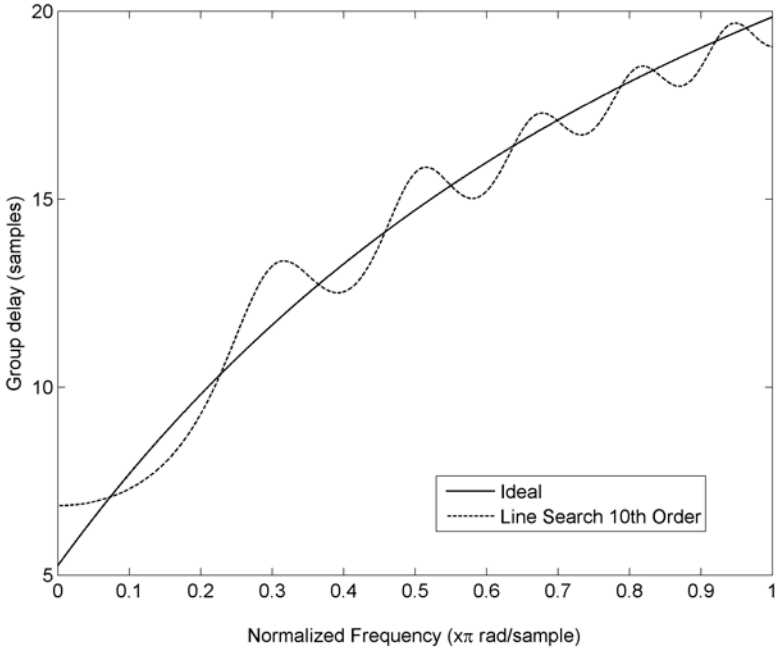


Figure 4.9 *Line Search 10th order filter group delay.*

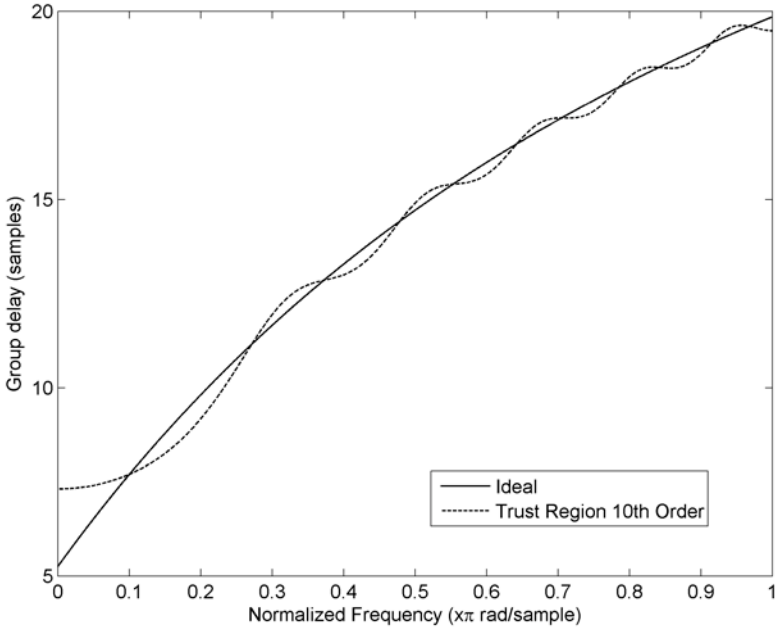


Figure 4.10 *Trust Region 10th order filter group delay.*

In all the designs, we see that the error is greater at the lower frequency bands. We could not readily explain why the error behaves in this manner. In an effort to mitigate this behavior, we weighted the lower frequencies so that they contributed relatively more to the objective function. This did not produce better fit of the group delay waveform.

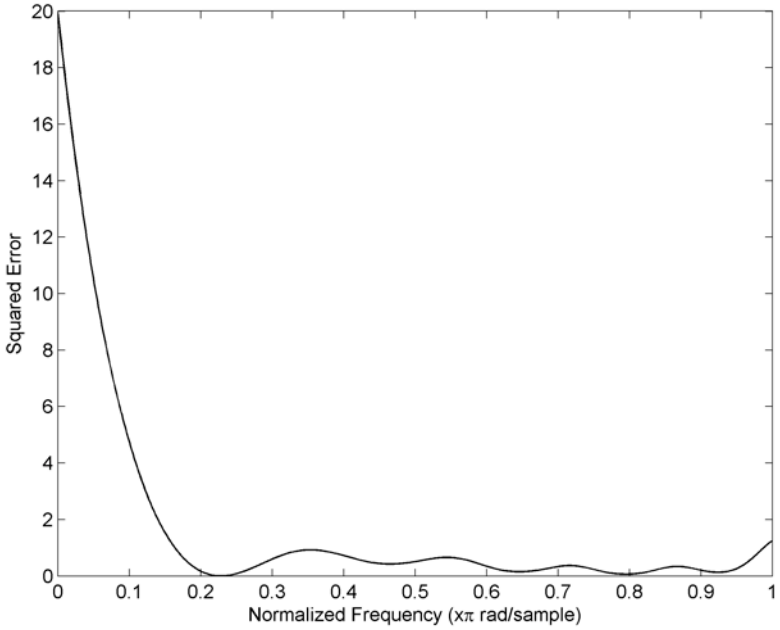


Figure 4.11 *Abel-Smith 10th order squared error.*

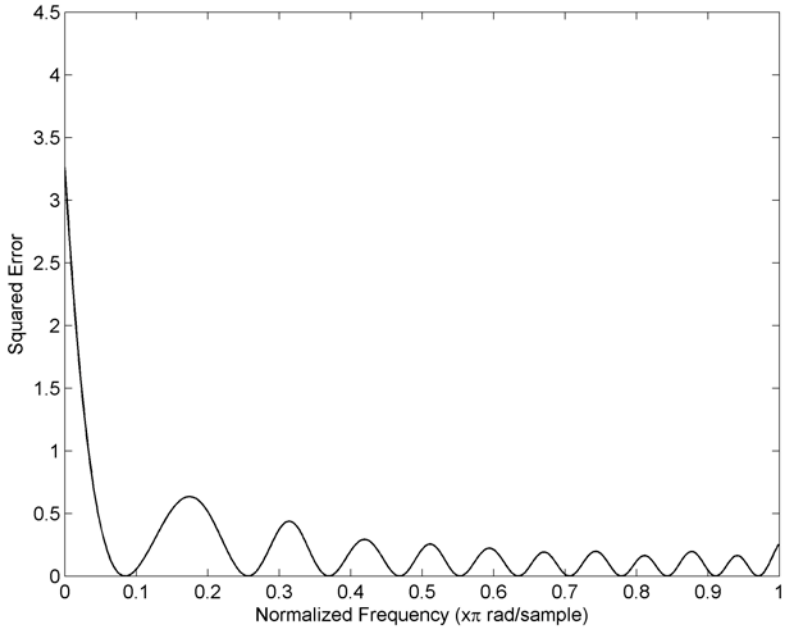


Figure 4.12 *Line Search 10th order squared error.*

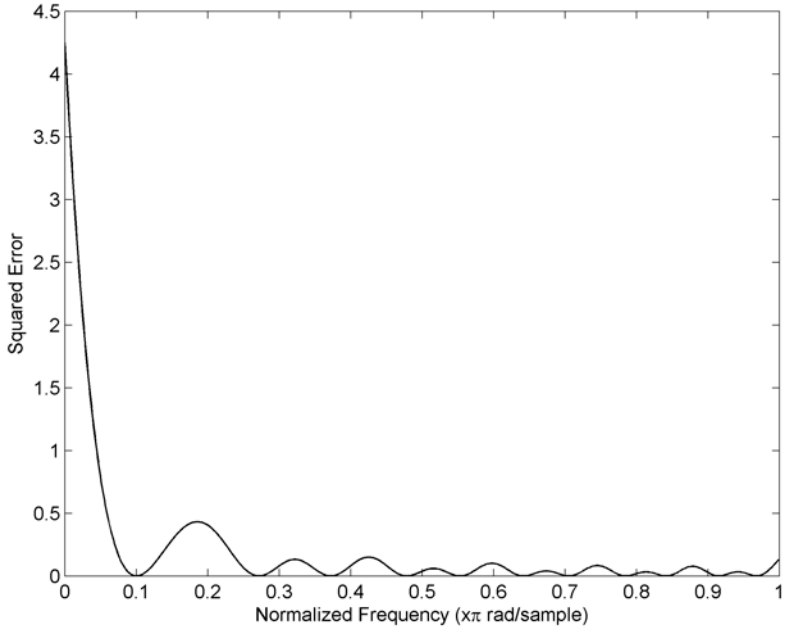


Figure 4.13 *Trust Region 10th order squared error.*

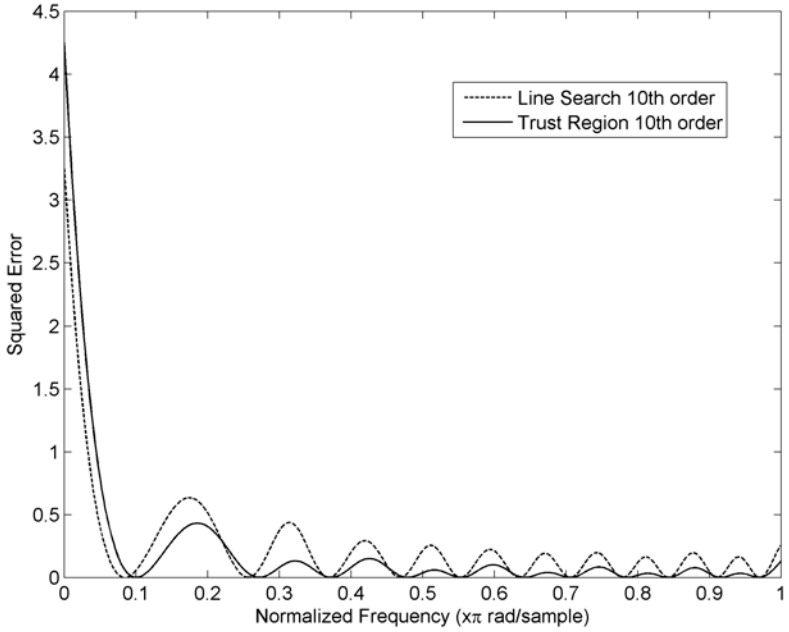


Figure 4.14 *Trust Region vs. Line Search 10th order.*

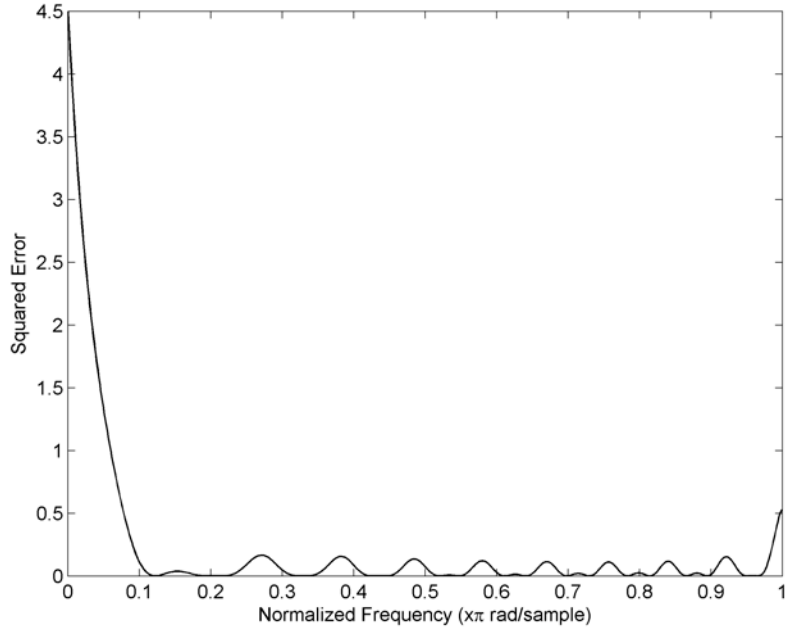


Figure 4.15 *Abel-Smith 20th order squared error.*

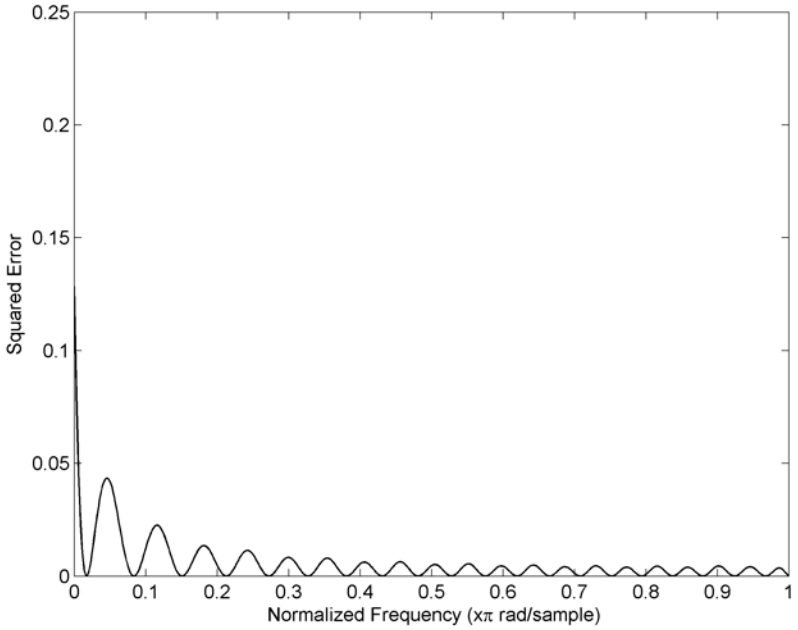


Figure 4.16 *Line Search 20th order squared error.*

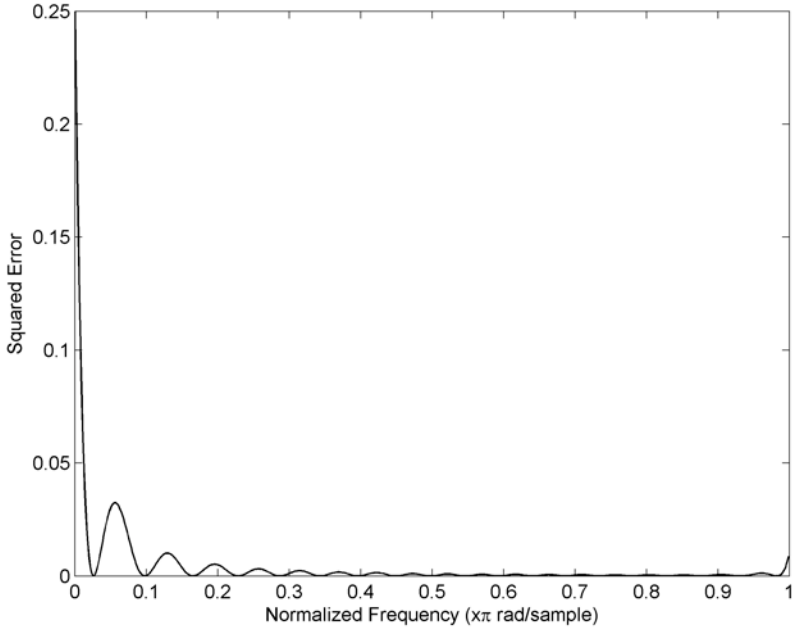


Figure 4.17 *Trust Region 20th order squared error.*

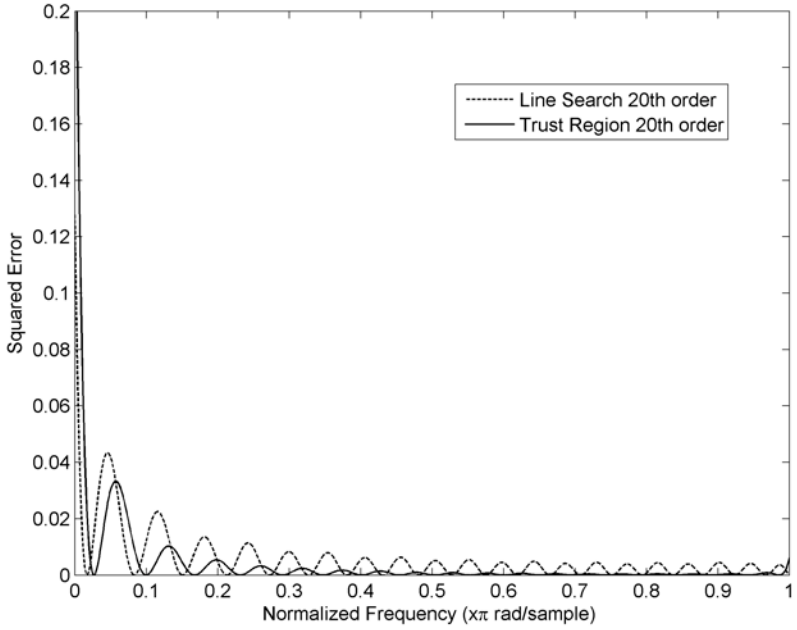


Figure 4.18 *Trust Region vs. Line Search 20th order.*

Viewing Figure 4.14 and Figure 4.18, we can see that the Line Search method, compared with the Trust Region method, produces less error in the lower frequencies, but more error in the higher frequencies. The Trust Region method produces lower error over such a large portion of frequencies that the resultant MSE is somewhat less than that of the Line Search method.

We also computed MSE values over a few different filter orders. As expected, the error is greatly decreased as the filter order rises. However, the slope of the plot is greatest between the 8^{th} and 10^{th} orders. The plot also demonstrates that increasing the filter order above 20 will result in very small decreases in the error.

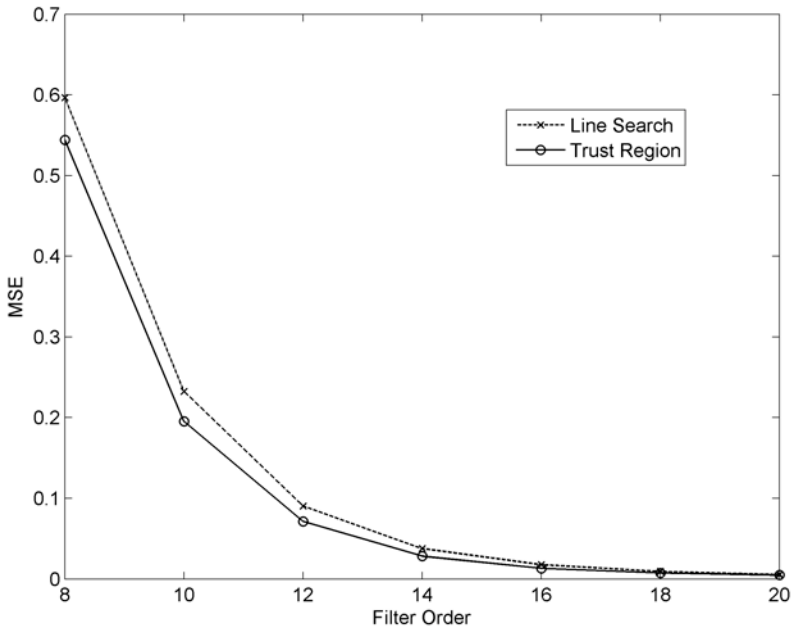


Figure 4.19 *MSE vs. filter order.*

4.3 Convergence of the Design

Any time an optimization routine is performed, there is concern over convergence of the design. Figure 4.20 demonstrates that the Trust Region method consistently converges to the same MSE over 50 trials that were undertaken. This behavior was demonstrated even with a random number generator supplying the initial guess for variable vector *x*.

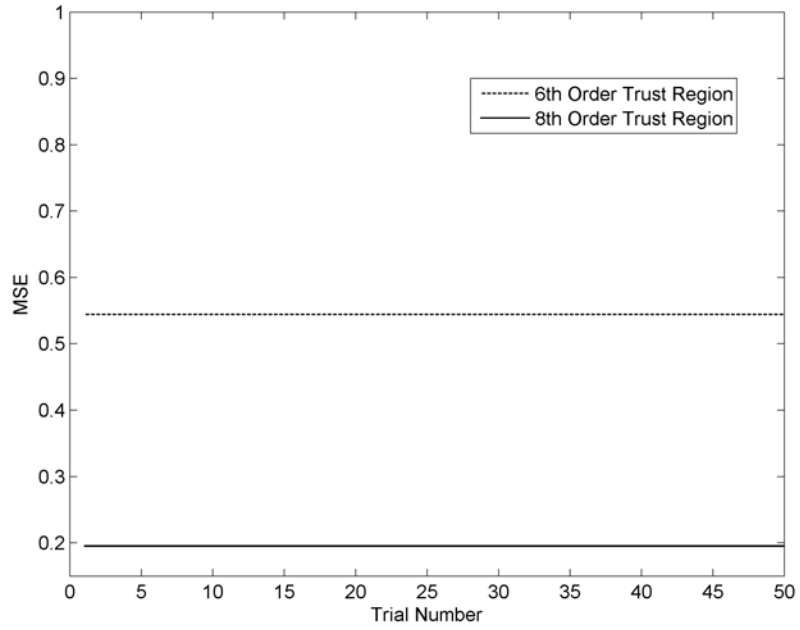


Figure 4.20 *MSE convergence over multiple trials.*

In Figure 4.21, we have plotted a pole-zero map for a 10^{th} order filter, and overlaid it a total of 20 times. We can see that the zeros and poles from each trial are nearly aligned. Figure 4.22 shows a zoomed in view of one pole cluster. We can see the difference between the largest pole radius and the smallest pole radius is only 0.33%. This indicates that the optimization produces essentially the same filter consistently even with a random initialization. This is important because it shows that we do not have to

supply an initialization vector that is a good guess. We can simply enter a random vector and be assured of convergence.

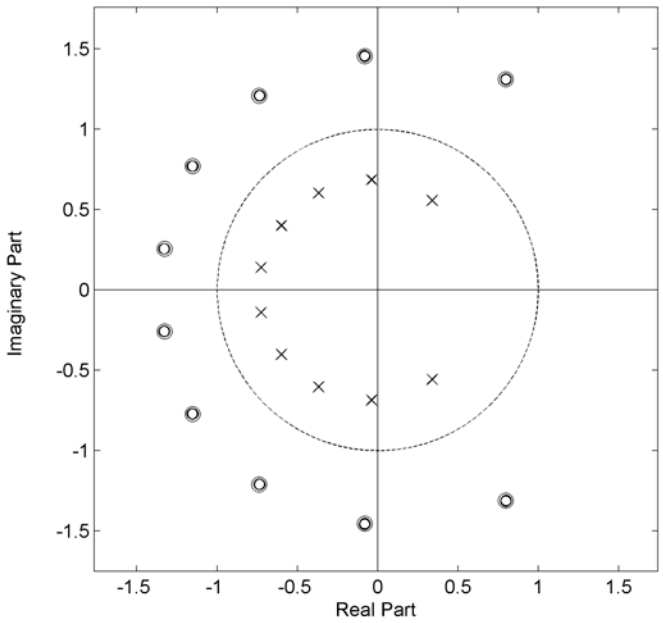


Figure 4.21 *10^{th} order pole-zero plot, overlaid for 20 trials.*

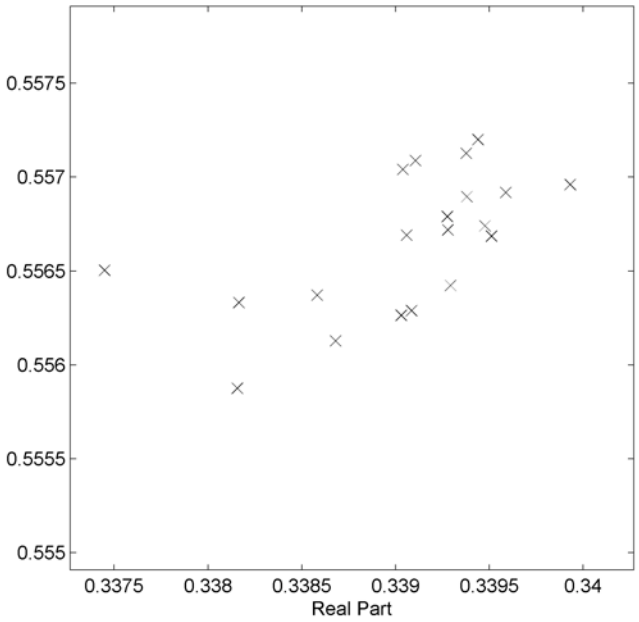


Figure 4.22 *Zoomed view of pole cluster for 20 trials.*

Although we showed that the Trust Region method converges consistently to the same MSE even with random initialization, there was an advantage in using the Abel-Smith filter pole locations as an initial guess. This behavior is shown in Figure 4.23, where there is a consistently lower number of function evaluations when initializing with the Abel-Smith vector vs. a random vector. The effect is much less pronounced at the lowest and highest filter order. The plot was made by averaging the number of function evaluations over 20 trials for each filter order.

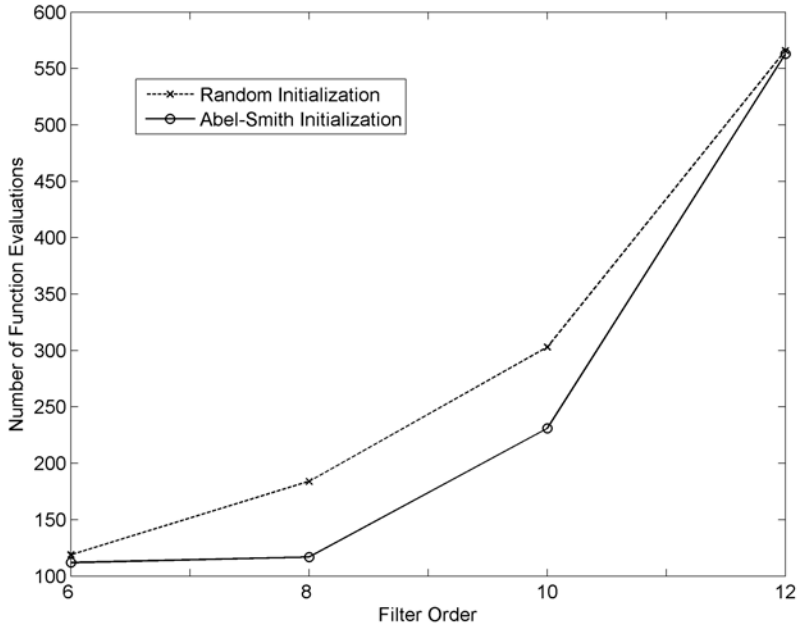


Figure 4.23 *Function evaluations for different initializations.*

To demonstrate the generality of the optimization method, Figure 4.24 shows the fit obtained with an arbitrary piecewise-linear ideal group delay waveform. Even with an 8^{th} order filter, the fit is close.

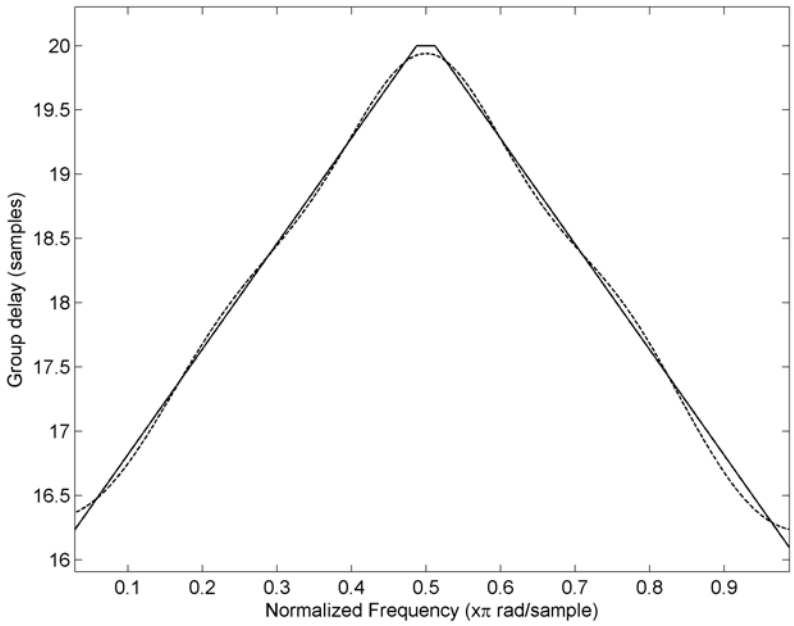


Figure 4.24 *8th order trust region matching of arbitrary waveform.*

Finally, we show in Figure 4.25 a dispersed pulse envelope, and the compression of that pulse by a 10th order filter. The pulse was generated with a TEC value of 2, so the second cascade stage produces the greatest output. The third stage is over doing the compression, and the output begins to drop off.

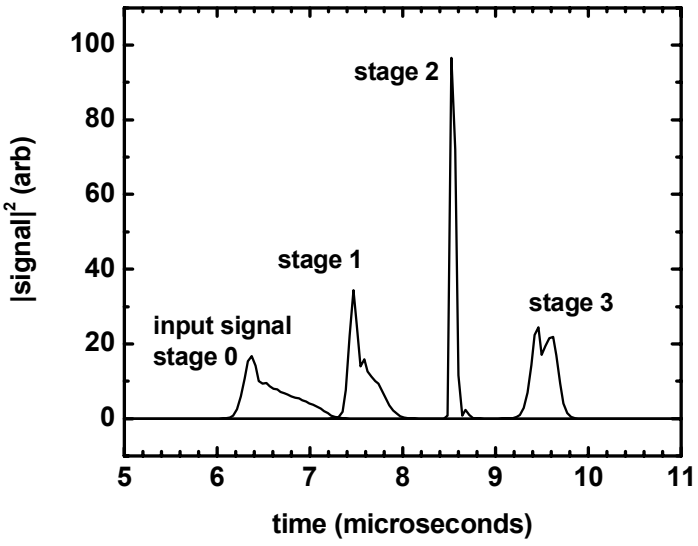


Figure 4.25 *10th order Trust Region filter compression.*

One final experiment we performed was aimed at improving the fit at the lower frequencies by using a weighting method. Basically, we modified the objective function in the following manner:

$$f_{err}(x) = \sum_{i=1}^{L} (w(\tau_{ideal,i} - \tau_{opt}(x)_{,i}))^2 ,$$

where w is the weight vector. The weighting vector simply puts more emphasis on certain frequencies relative to the other frequencies. Since we wished to emphasize the lower frequency fit, we chose a piecewise constant weight vector:

$$w = [1 \quad 1 \ldots 1 \quad 0.9 \quad 0.9 \ldots 0.9] .$$

So, in this example, we are weighting the higher frequencies only 90% relative to the lower frequencies. This is clear in the plot of Figure 4.26.

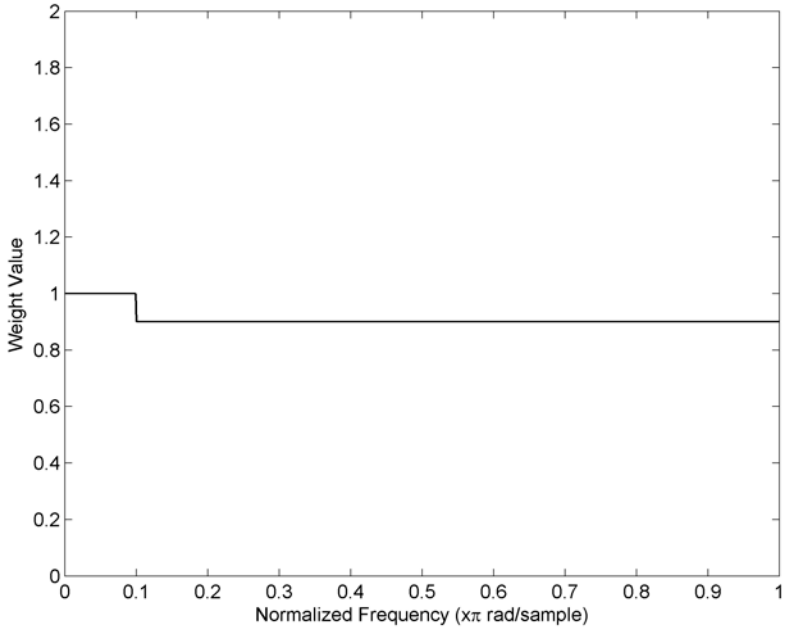


Figure 4.26 *Piecewise constant weight vector.*

We expected this weighting vector added to the optimization routine would improve the lower frequency fit. The results we obtained did not support that hypothesis. Figure 4.27 shows the original unweighted optimization results, and Figure 4.28 shows the results with the weight vector of Figure 4.26 applied. Clearly, the fit is not better at the lower frequencies; in fact it has worsened somewhat. We believe this is because the un-weighted solution is the global minimum.

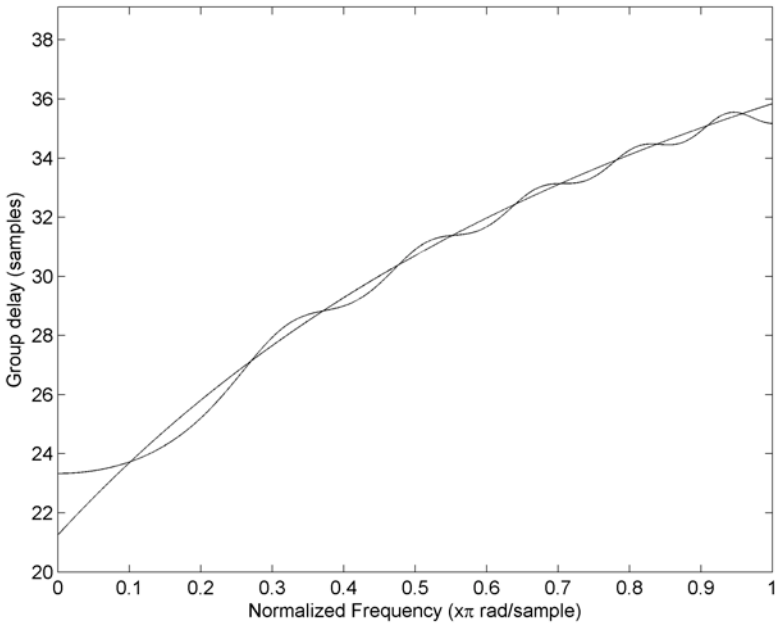


Figure 4.27 *Original unweighted 10th order filter fit.*

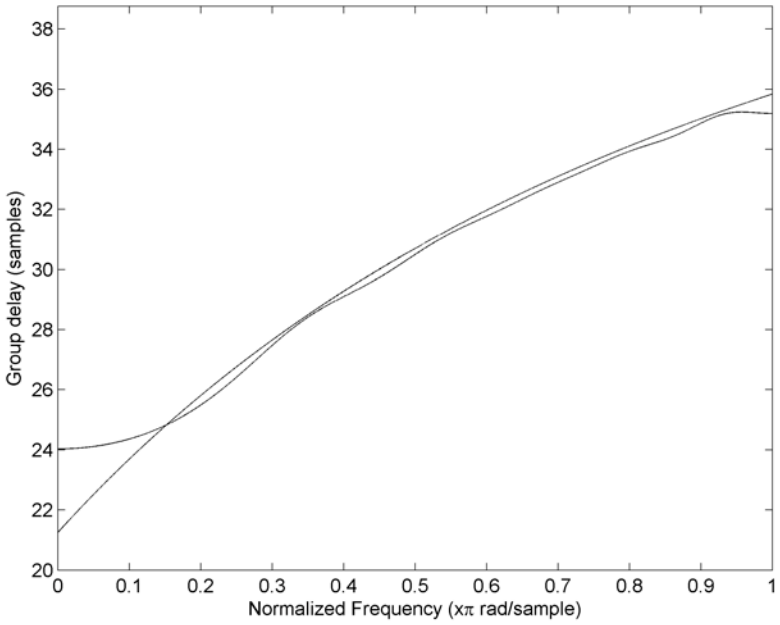


Figure 4.28 *Weighted 10th order filter fit.*

We have demonstrated some of the results obtained by the various filter design approaches. We are ready now to draw some conclusions and also to look at some of the next steps that can be taken to improve what we have done.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, we described the dispersion problem facing space-borne pulse detection systems. The Appleton-Lassen equation was introduced as a model for the behavior of the ionosphere. The system model emphasized that the ionosphere mainly affects the phase of the signal passing though it. Some previous pulse detection projects were mentioned, where we pointed out that these approaches were analog designs. Some of the advantages of a digital system were discussed. We sought a digital filter that could accomplish an anti-dispersion equalization of the ionosphere transfer function.

We changed the formulation slightly so that we approached it as a group delay matching problem. Group delay equalization proved to be an effective design approach to the anti-dispersion filter. Although a true matching would require a negative group delay, we showed that the group delay waveform could simply be shifted by an arbitrary constant. The only penalty for this is a constant group delay equal to the amount of offset. For our purposes, this constant group delay was not detrimental to the goal of pulse compression.

The Abel-Smith algorithm entered the discussion as a simple and reliable means of designing the compression filter. To set the stage for the Abel-Smith filter, we described some useful characteristics of allpass filters. We showed that the Abel-Smith approach used a cascade of bi-quad allpass sections to create the desired group delay characteristic. After deriving the ionosphere group delay behavior, we were able to create Abel-Smith filters of arbitrary order that effectively matched the group delay waveform.

To improve upon the Abel-Smith design, we shifted our focus to an optimization approach. We retained the cascaded allpass design, but effectively performed a unique design for each section, vs. the uniform method Abel-Smith used. This greatly improved the group delay match that the filter was able to achieve. Abel-Smith used very high filter orders (e.g., 128^{th} order) to achieve good matching characteristics. The optimized methods yielded filters with lower error than the Abel-Smith filter while requiring only approximately half the coefficients. The optimized filter designs proved to be an effective way to create anti-dispersion filters. We should note that the difference between optimizations routines was small compared with the difference between optimizing and not optimizing. For any group delay equalization process with limited filter order, our first choice should be an optimized filter. Clearly, it was worth the effort to optimize the filter because the resulting fits are substantially better.

It was surprising to discover that adding a weight vector to the optimization routine did not improve the fit of the function. We found through experimentation that

using a weight vector can be tricky. We will leave it to future work to crystallize a method of using a weight vector more effectively.

5.2 Future Work

There are a number of different paths that could be beneficial for future work. One avenue that could be explored is the design of a filter having complex coefficients. This would abandon the use of biquad sections in the optimized filter, while retaining the allpass filter trait. It is possible that by removing the restriction of always having conjugate poles, that a better group delay fit may be achieved. Another practical extension would be to carry out the approximations made in the ionosphere model to higher orders. This would create a more accurate ionosphere group delay function. It is irrelevant how precisely a waveform can be matched if that waveform is not an accurate representation of the system behavior. In addition, the sampling process could be modified to create a quadrature sampled signal. The use of a quadrature sampling scheme would reduce the sample rate by a factor of two. Although this would not reduce the data rate, if there is a restriction on the A/D clock speed it may be beneficial.

Appendicies

Appendix A: MATLAB Scripts

Appendix B: Direct Form Transfer Function

Appendix C: Least p-th Algorithm

Appendix D: Trust Region Algorithm

Appendix E: Misc. Signal Processing

Appendix A: MATLAB Scripts

Ionosphere dispersion function:

```
% ionodisp: dispersion funtion for the ionosphere model
function [d,f]= ionodisp(TEC, flo, fhi, nyquist,npts)
% ;INPUTS:
% ;   TEC in TECU
% ;   flo in MHz low end of frequency range
% ;   fhi in MHz high end of frequency range
% ;   npts number of points in the full complex frequency-
domain waveform
% ;      if unsure, enter a small number and the program
will suggest
% ;      a minimum.  Typically a value greater than the
minimum that is
% ;      an integral power of 2 is chosen.
% ;   fNyquist in MHz Nyquist limit to be used in creating
the frequency
% ;      domain spectrum.    The waveform has an equivalent
sampling rate
% ;      of 2 * fNyquist MHz.  fNyquist must be greater
than fhi.
% ;OUTPUTS:
% ;   d:returns a complex array with the baseband
frequency-domain transfer
% ;   function of the dispersed ionosphere waveform in FFT
wrap around format.
% ;   ifft of the result gives the time-domain dispersed
impulse.
% ;   The time domain waveform
% ;   has the time origin in the center of the pulse to
minimize oscillations
% ;   in the frequency-domain representation.  The
frequency-domain signal
% ;   has been scaled to give order-unity time-domain
amplitudes.  Specifically,
% ;   the total of the square of the time-domain pulse will
yield the
% ;   number of points in the array.  This allows
comparisons between
% ;   amplitudes of waveforms with different TEC values for
the same bandwidth
% ;   and number of points.

% ;   f: frequency axis that can be used to plot the
frequency-domain
```

```
% ;   waveform.  It may also be used to generate a
butterworth filter
% ;   transfer function using the function call
hbutter7pass(f,f0,bw) or
% ;   hbutter7lohi(f,flo,fhi).  The time domain interval is
1/(N*f[1]).
% ;METHOD:
% ;The transfer function is based on a dispersion term for
which the
% ;time origin has been shifted to the middle of the
dispersed pulse
% ;in order to minimize the number of oscillations that
have to be
% ;represented in the frequency domain.  The time origin
shift is based
% ;on the simple free-electron plasma model for the
ionosphere.
% ;Derivation of the formulas are given in "Simple Model
for Pulse
% ;Propagation Through the Ionosphere".
%
% ;The model assumes a slab equivalent of 400 km thickness
% ;for a given TEC in units of 1e16 e/m^2.  The slab
thickness
% ;is a means to an end because it effectively cancels out
% ;during the calculation, leaving only the TEC value.
% ;
%
% ;
% ;The output is a frequency-domain representation of the
dispersed
% ;signal in FFT wrap around format.  The Nyquist limit is
typically
% ;set as some multiple of the GPS clock, 10.23 MHz for the
kinds of
% ;problems that we are currently working on.

    fpsq = 2.0154096 * TEC; %slab model plasma frequency
squared
    fp = sqrt(fpsq); %causes a zero in the refractive index
%disp(['plasma frequency: ' num2str(fp)]);
    if fp >= flo
        disp(['ERROR: all frequencies must be larger than '
num2str(fp)]);
        return
       end
    if fhi >= nyquist | flo >= nyquist
```

```
              disp('ERROR: flo and fhi must be less than
fNyquist');
        return
       end

    tauX = 1334.25638;   %X/c also called tau sub x
%determine frequency interval from optimized time origin

    nbarlo = sqrt(1.0 - fpsq /( flo^2)); %refractive index
equation
    nbarhi = sqrt(1.0 - fpsq /( fhi^2));
    range = tauX * (1.0 / nbarlo - 1.0 / nbarhi);
        %range is the range of delay times expected for the
given
        % flo to fhi

%determine minimum number of points in frequency array
%
    Delta = 0.5 / range; %k
    m = fhi / Delta;
    m = 2 * m;%to guarantee an even number of points
    if npts < m
        disp(['Error: need at least ' num2str(m)  '
points']);
        return
     end
    m=npts/2;
    Delta = nyquist / m;%frequency bin spacing
      %disp(['Sampling frequency is ' num2str(2*nyquist)  '
MHz']);
%
%
    f=Delta*[0:m];%extent of the frequency vector up to the
Nyquist frequency
    f=[f,-fliplr(f(2:m))];%generate frequency scale in
wraparound order.  This doubles size
    idx = find(abs(f) >= flo & abs(f) <= fhi);

    count=length(idx);
     %disp([ num2str(count) ' points dispersion in total of
' num2str(2*m)]);

   %calculate the function

    a = 0.5 * (1/nbarlo + 1/nbarhi);
        taud = tauX * a;% the average dispersed delay time,
including
```

```
        %vacuum propagation through X
    tau_shift = tauX * a - tauX;
        %origin with respect to vacuum arrival through all
x
    nbar = sqrt(1 - fpsq./ (f(idx)).^2);
    T = tauX * nbar - taud;
        %a shifts time origin to mid-range of delays
    d=complex(zeros(1,2*m));
    length_d=length(d);
    d(idx)=exp(complex(0,-2*pi*T.*f(idx)));
    %d=d/sqrt(count);  %roughly normalize output
end
```

Abel-Smith IIR Allpass Equalizer Function:

```
%this function implements the Abel-Smith algorithm
 function iircoeff=abelsmith(N,beta)
%choose the compensator order, this is the number of
sections in the
%cascade, which is equal to how many sub-bands are
partioned out of the
%desired group delay function
%At a minimum, N=5 so the group delay is everywhere
positive in our band of interest
%create the group delay waveform for the compensator
v=linspace(0,1,512); %create the frequency range vector
b=1344.5;%the b constant
fs=51.15;%the sampling frequency
b_hat=b/fs;%^the normalized b unit
tau_compen=2*N+2*b_hat/3-b_hat./((v+1).^2);
%tau_compen=27.5233-b_hat./((v+1).^2);
plot(v(end/2+1:end),tau_compen(end/2+1:end));%plot the
waveform
```

```
%plot(v,tau_compen)
title('Desired Compensator Group Delay')

%now, we need to partition the waveform into sub bands
%to do this, we call the band_edge_finder function
t0 = (2*N+2*b_hat/3);

e(1)=0;
for i=2:N+1
 e(i)=  (-(t0-t0*e(i-1) -b_hat/(e(i-1)+1)-1)...
        +sqrt((t0-t0*e(i-1) -b_hat/(e(i-1)+1)-1).^2 ...
        -4*t0*(b_hat-t0*e(i-1) -b_hat/(e(i-1)+1)-
1)))/(2*t0);
end

edge_list=e;
%use the abel smith algorithm to get the coefficients for
each section, where
%each section is of single order

iircoef = complex(zeros(2,N,2));%initialize it to zero
fit = zeros(1,512/2);%this is the cumulative effect of all
the sections, creating the overall group delay waveform
%it is this "fit"  waveform that is being matched to
"tau_compen"

for i=2:N+1
    omega_pole=(edge_list(i)+edge_list(i-1))/2;%find the
band center frequency
    delta=(edge_list(i)-edge_list(i-1))/2;%find the
bandwidth of that particular sub band
    eta=(1-beta*cos(delta))./(1-beta);
    rho=eta-sqrt(eta.^2-1);%find the radius of the pole
    b_section(i-1,:)=[-rho*exp(-j*omega_pole) 1];%the
numerator coefficients for the section
    a_section(i-1,:)=[1 -rho*exp(j*omega_pole)];%the
denominator coefficients for the section
    iircoeff(i-1,:,1)=b_section(i-1,:);
    iircoeff(i-1,:,2)=a_section(i-1,:);
    tau_section=(1-rho.^2)./(1+rho.^2-2*rho.*cos(2*pi*(v-
omega_pole)));%the group delay function of the section
fit=fit+fliplr(tau_section(512/2+1:512))+tau_section(1:512/
2);
    figure;plot(v,tau_section)
end

%figure;plot(v(1:512/2),fit)
```

Iirgrpdelay fitting script:

```
v=linspace(-.5,.5,4*512); %create the frequency range
vector
b=1344.5;
fs=51.15;%the sampling frequency
b_hat=b/fs;%^the normalized b unit
ideal_group_delay=2*N+2*b_hat/3-
b_hat./((v(end/2+1:end)+1).^2);


f=linspace(0.1,1,100)%create a frequency vector over the
desired range
%gd=16*12.915-4*.12915./(f.^2);%create the desired group
delay
N=10; %filter order
gd=25-1*.12915./(f.^2);%create the desired group delay
[num,den]=iirgrpdelay(N,f,[f(1) f(end)],gd) %create a
filter that will exhibit the desired group delay
[ga,f2]=grpdelay(num,den,100,2)% get the actual group delay
the filter has

offset=gd(1)-ga(10);%the offset is the desired group delay
minus the actual group delay
x=ga+offset;

close all;
plot(f2,x)  %plot the actual group delay of the synthesized
filter with offset in BLUE
hold
plot(f,gd,'r')%plot the desired, or target, group delay in
RED
```

Optimization routine use optimization toolbox:

```
%creates an optimum IIR filter that checks for stability in
the transfer
%function.  It matches an arbitrary group delay, and uses
the values from
%the Abel-Smith design as an initial guess. It assumes the
filter is built
%of a cascade of allpass biquad sections

clear all
```

```
global ideal_delay

N =5 ; %the number of biquad sections
npts=2*512; %the number of points to use in group delay
plots
offset=20;
if 0 %create the Abel-Smith filter

    beta = .8;
    %the fraction of the waveform present at the band
edges.
    %High beta means a sharper peak, low beta means a
smoother peak.


    %an offset used for the group delay waveform to account
for the delay
    %caused by the filtering process. It is arbitrary for
our purposes.


[cascade_b_coeff,cascade_a_coeff,overall_b_coeff,overall_a_
coeff]=abelsmith(N,beta);
    %creates the actual Abel-Smith filter. cascade_b_coeff
and
    %cascade_a_coeff are matrices with each row containing
one biqaud
    %section's coefficients.  The overall coefficients are
the result
    %of convolving all these sections together.

    abel_smith_delay=grpdelay(overall_b_coeff,
overall_a_coeff, npts);
    %creates the group delay of the Abel-Smith filter

    ideal_delay=ideal_group_delay(N)+offset;
    %ideal_delay=[2*linspace(8,10,500) 20*ones(1,24)
2*linspace(10,8,500)];
    %creates the delay waveform we are trying to match.
Includes an
    %offset that is arbitrary

    figure;set(gca,'Fontsize',12)
    plot(linspace(0,1,npts),ideal_delay, 'k');
    hold on;
    plot(linspace(0,1,npts),abel_smith_delay+offset, 'k');
```

```
    ylabel('Group delay (samples)','FontSize',12)
    xlabel('Normalized Frequency (x\pi
rad/sample)','FontSize',12)
    %title(['Ideal vs Abel-Smith design with N=' num2str(N)
' and \beta=' num2str(beta)])
    %legend('Ideal',['Abel-Smith ' num2str(2*N) 'th
Order'],'Location','SouthEast');
    %plot the matching of the group delay waveforms

    figure;freqz(overall_b_coeff,overall_a_coeff)
    %plot the frequency response of the filter to check for
allpass
    %behavior

    figure;zplane(overall_b_coeff,overall_a_coeff)
    %plot the zplane map to check for stability of the
poles

    disp('Mean square error for Abel-Smith filter:');
    mean((ideal_delay(:) - abel_smith_delay(:)-offset).^2)
    %disp the MSE to see how close the fit actually is

end


if 0 %create the filter using iirgrpdelay.m in the filter
design toolbox
    [num,den] = iirgrpdelay(N,linspace(0,1,npts),[0
1],ideal_delay+offset);
    figure;grpdelay([0 0 0 npts],den);hold
on;plot(linspace(0,1,1024),ideal_delay+offset,'k');
    grid off;
    %title('Matlab iirgrpdelay function 10th order')
end


if 1 %create the optimized filter
    ideal_delay=ideal_group_delay(N)+offset;
    x0 = abs(rand(2*N,1));
    %choose a random vector to initialize the optimization
routine
    %length is 2N because we have N pole radii and N pole
frequencies

    %ideal_delay=[2*linspace(8,10,500) 20*ones(1,24)
2*linspace(10,8,500)];
    %ideal_delay=20*ones(1,1024);
```

```
    %[x,resnorm,residual,exitflag,output,lambda,jacobian] =
lsqnonlin(@grp_diff_stable, x0,0*[1 1 1 1 1 1 1 1],[1 1 1 1
1 1 1 1]);%here is where you would put lower and upper
bounds

    [x,resnorm,residual,exitflag,output,lambda,jacobian]
=lsqnonlin(@grp_diff_stable, x0);%here is where you would
put lower and upper bounds


          %x0 = x;
    %[x,resnorm,residual,exitflag,output,lambda,jacobian]
=lsqnonlin(@wgrp_diff_stable, x0);

    % Solution
    Num = 2*512;
    radii_final=x(1:end/2);%get the optimized pole radii
    thetas_final=x(end/2+1:end);%get the optimized pole
angles



    poles_final=[radii_final.*exp(j*2*pi*thetas_final);
radii_final.*exp(-j*2*pi*thetas_final)];%form the optimized
poles and zeros

zeros_final=[(1./radii_final).*exp(j*2*pi*thetas_final);
(1./radii_final).*exp(-j*2*pi*thetas_final)];

[num_final,den_final]=zp2tf(zeros(length(poles_final),1),po
les_final,1);%form a temporary transfer function with a
real denominator, temp numerator
    %num_final=num_final*den_final(end);%normalize the gain
of the transfer function

    num_final=fliplr(den_final);
    final_soln = grpdelay(num_final, den_final, Num);

    %final_soln=grpdelay(overall_b_coeff, overall_a_coeff,
Num);
    % Delay?
    opt_delay = (mean(ideal_delay(:) - final_soln(:)))
    figure;
    plot(linspace(0,1,npts),ideal_delay, 'k'), hold on
    hold on;
    plot(linspace(0,1,npts),final_soln+opt_delay, 'k')
```

```
    %title('original')
    axis([0 1 20 1.1*max(final_soln+opt_delay)])
    xlabel('Normalized Frequency (x\pi
rad/sample)','FontSize',12);
    ylabel('Group delay (samples)','Fontsize',12);
    %disp('Mean square error final with optimal extra
delay:');
    %mean((ideal_delay(:) - final_soln(:) - opt_delay).^2)

    %figure;plot(mean(ideal_delay(:) - final_soln(:) -
opt_delay).^2,'k')
    %opt_b_coeff=[zeros(1,opt_delay) x];
    %opt_a_coeff=[fliplr(x) zeros(1,opt_delay)];
    %toc
    %figure;pzmap(num_final,den_final);title(['N='
num2str(N) ' and \beta=' num2str(beta)])
    %%figure;freqz(num_final,den_final)
end
```

Objective function called by lsqnonlin.m:

```
function [err_diff] = grp_diff_stable(B)
global ideal_delay

Num = 2*512;
radii=B(1:end/2);
thetas=B(end/2+1:end);

poles_first_order=[radii.*exp(j*2*pi*thetas)];%make the
first order poles

poles_biquad=[poles_first_order; conj(poles_first_order)];
%add in the conjugate poles for biquads


zeros_first_order=[(1./radii).*exp(j*2*pi*thetas)];%make
the first order zeros
zeros_biquad=[zeros_first_order;
conj(zeros_first_order)];%add in the conjugate zeros for
biquads


%[num_temp,den_working]=zp2tf(zeros(length(poles_biquad),1)
,poles_biquad,1);%using the zeros and pole, form the
transfer function
```

```
[num_working,den_working]=zp2tf(zeros(length(poles_biquad),
1),poles_biquad,1);%using the zeros and pole, form the
transfer function
%num_working=num_working*den_working(end);
num_working=fliplr(den_working);
overall_group_delay = grpdelay(num_working,den_working,
Num);
%overall_group_delay =
grpdelay(fliplr(den_working),den_working, Num);


w=ones(1,1024);
%w=([2*ones(1,100) 1*ones(1,1024-100)]);
%w=fliplr(w);
%w=1;
%w=exp(-(linspace(0,1.2,1024)).^2);
opt_delay = (mean(ideal_delay(:) -
overall_group_delay(:)));
err_diff = w(:).*(ideal_delay(:) -
overall_group_delay(:));%+ opt_delay);
return
```

Appendix B: Direct Form Transfer Function

The time domain output sequence y_k of an Lth-order recursive filter with input sequence x_k is given by

$$y_k = \sum_{n=0}^{L} b_n x_{k-n} - \sum_{n=1}^{L} a_n y_{k-n} \quad \text{(A-1)}$$

The question may be asked: what is $H[z]$, the transfer function of the filter?

From the definition of the z-transform of a sequence s_k, which is length N:

$$S[z] = \sum_{k=0}^{N-1} s_k z^{-k}$$

Let us take the z-transform of both sides of equation (A-1):

$$Z\{y_k\} = Z\left\{\sum_{n=0}^{L} b_n x_{k-n} - \sum_{n=1}^{L} a_n y_{k-n}\right\}$$

Expand the summation, and apply the definition of the transform:

$$Y[z] = \sum_{k=0}^{N-1}\left[\left(\left(b_0 x_k + b_1 x_{k-1} + \cdots + b_L x_{k-L}\right) - \left(a_1 y_{k-1} + a_2 y_{k-2} + \cdots + a_L y_{k-L}\right)\right) z^{-k}\right]$$

Distribute the z term into the summation argument:

$$Y[z] = \sum_{k=0}^{N-1}\left[\left(b_0 x_k z^{-k} + b_1 x_{k-1} z^{-k} + \cdots + b_L x_{k-L} z^{-k}\right) - \left(a_1 y_{k-1} z^{-k} + a_2 y_{k-2} z^{-k} + \cdots + a_L y_{k-L} z^{-k}\right)\right]$$

Now, we recognize that we can split up the summation of a sum into a sum of summations:

$$\begin{aligned} Y[z] = {} & b_0 \sum_{k=0}^{N-1} x_k z^{-k} + b_1 \sum_{k=0}^{N-1} x_{k-1} z^{-k} + \cdots + b_L \sum_{k=0}^{N-1} x_{k-L} z^{-k} \\ & - a_1 \sum_{k=0}^{N-1} y_{k-1} z^{-k} - a_2 \sum_{k=0}^{N-1} y_{k-2} z^{-k} - \cdots - a_L \sum_{k=0}^{N-1} y_{k-L} z^{-k} \end{aligned}$$

Now, to continue we need to use a property of the z-transform known as the delay property:

If

$$s_k \Leftrightarrow S[z]$$

Then

$$s_{k-m} \Leftrightarrow z^{-m} S[z]$$

Using this substitution, we have:

$$Y[z] = b_0 X[z] + b_1 z^{-1} X[z] + \cdots + b_L z^{-L} X[z] - a_1 z^{-1} Y[z] - a_2 z^{-2} Y[z] - \cdots - a_L z^{-L} Y[z]$$

Now, if we pull the $X[z]$ and $Y[z]$ terms out of the sum:

$$Y[z] = X[z](b_0 + b_1 z^{-1} + \cdots + b_L z^{-L}) - Y[z](a_1 z^{-1} + a_2 z^{-2} + \cdots + a_L z^{-L})$$

Grouping $Y[z]$, and forming the transfer function:

$$Y[z](1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_L z^{-L}) = X[z](b_0 + b_1 z^{-1} + \cdots + b_L z^{-L})$$

$$H[z] = \frac{Y[z]}{X[z]} = \frac{b_0 + b_1 z^{-1} + \cdots + b_L z^{-L}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_L z^{-L}}$$

The vectors *b* and *a* can be formed from these numerator and denominator coefficients:

$$b = [b_0 \; b_1 \cdots b_L]$$
$$a = [1 \; a_1 \cdots a_L]$$

Appendix C: Least p-th Algorithm

1. Input x_0 and ε_1. Set $k = 1$, $p = 2$, $\mu = 2$, $E_0 = 10^{99}$.
2. Initialize frequencies $\omega_1, \omega_2, \ldots, \omega_K$.
3. Using x_{k-1} as initial value, minimize

$$\Psi_k(x) = E(x)\left\{\sum_{i=1}^{K}\left[\frac{|e_i(x)|}{E(x)}\right]^p\right\}^{1/p}$$

where

$$E(x) = \max|e_i(x)|$$

with respect to x, to obtain x_k. Set $E_k = E(x)$.

4. If $|E_{k-1} - E_k| < \varepsilon_1$, then output x_k and E_k, and stop. Otherwise, set $p = \mu p$, $k = k + 1$ and go to step 3.

Appendix D: Trust Region Algorithm

The optimization algorithm that was used is called the interior trust region method. This is a method that employs constraints on the variables. The increment $s_k = x_{k+1} - x_k$ is an approximate solution to a quadratic subproblem with a bound on the step:

$$\min\left\{\psi_k(s) \equiv g_k^T s + \frac{1}{2} s^T (B_k + C_k)s : \|D_k s\| \le \Delta_k\right\} .$$

Here, $g_k \equiv \nabla f(x_k)$, B_k is a symmetric approximation to the Hessian matrix $\nabla^2 f(x_k)$, D_k is a scaling matrix, and Δ_k is a positive scalar representing the trust region size. Also, for a vector *y,* the 2-norm is defined $\|y\| = \sqrt{y^T y}$. Also, the variables are constrained so that $l \le x \le u$. Then, the algorithm defines a vector $v(x)$ as follows:

(i) *if* $g_i < 0$ *then* $v_i \equiv x_i - u_i$
(ii) *if* $g_i \ge 0$ *then* $v_i \equiv x_i - l_i$

For any vector *s,* diag(*s*) denotes an *n*-by-*n* diagonal matrix with *s* defining the diagonal entries. Using this notation:

$$D(x) \equiv \text{diag}(|v(x)|^{-\frac{1}{2}})$$

Appendix E: Miscellaneous Signal Processing

Bandpass Signal Sampling

Usually, the idea of aliasing invokes a negative picture of frequencies higher than the Nyquist frequency (half the sampling frequency) folding back and essentially contaminating the band below the Nyquist frequency. The process of undersampling can employ the phenomenon of aliasing to reduce the sampling frequency in a system. The process is also known as bandpass sampling, or super-Nyquist sampling, and involves using the Nyquist theorem to create a system configuration that aliases higher frequency signals that occur at greater than the ADC sampling rate. For a system using this technique, the signal of interest has a bandwidth *B* containing frequencies higher than the sampling frequency. Also, the bandpass signal has a center frequency, f_c, associated with it. An analog bandpass filter limits the bandwidth of the signal, and acts similar to an antialiasing filter. Two formulas are useful in finding the appropriate sampling frequency:

$$f_{sampling} > 2B$$

$$f_{sampling} = \frac{4f_C}{2Z-1}$$

where Z is a rounded down whole number. For example, if we use a bandwidth of 25.575 MHz, and a center frequency of 63.9375 Mhz, from the first equation the initial

estimate of the sampling frequency would be 51.15 MHz. Using this value for the sampling frequency in the second equation, we would determine that *Z* needs to be:

$$Z = \frac{1}{2}\left(\frac{4f_c}{f_{sampling}} + 1 \right) = \frac{1}{2}\left(\frac{4(63.9375)}{51.15} + 1 \right) = 3$$

Then, using this value of Z, we again calculate the sampling rate using the second equation as

$$f_{sampling} = \frac{4f_c}{2Z - 1} = 4(63.9375)/5 = 51.15 \text{ MHz}$$

In this case, the second calculation of the sampling frequency appeared to be redundant, but the second calculation usually results in a slightly different value than the first equation indicates. That is, often the number for Z would be perhaps 3.5, which would have been rounded down to 3.

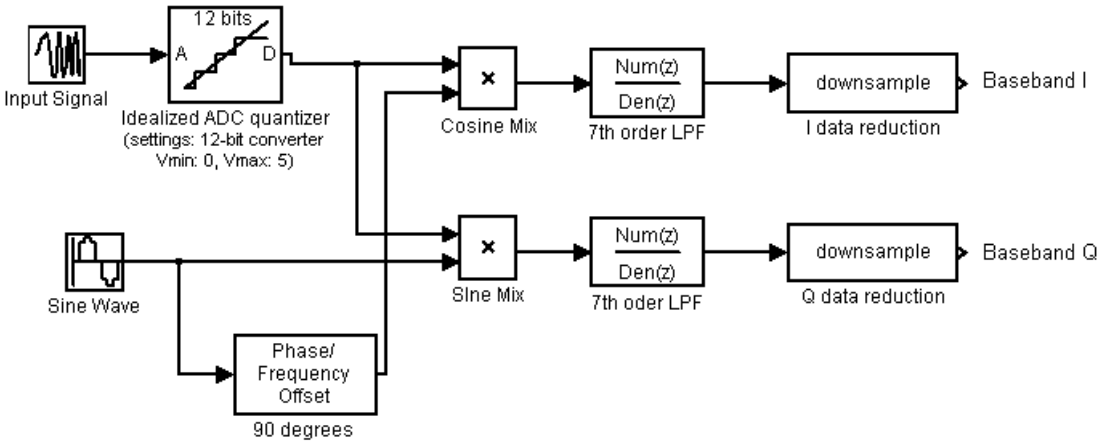
The idea here is that when a signal is sampled at less than the Nyquist rate, the aliased signal appears at $f_{sampling} - f_{signal}$. Actually, a more accurate formula is given by

$$Alias\ Frequency = \left| Closest\ Integer\ Multiple\ of\ Sampling\ Frequency - Input\ Frequency \right|$$

That is, a 410 MHz sine wave sampled at 50 MS/s will appear at | 400 MHz-410MHz |=10 MHz. The usefulness of the technique is shown for bandpass signals. If the signal of interest contains frequencies from DC to 40 MHz, then an ADC operating at twice the

bandwidth must be used, meaning 80 MS/s. Here, the bandwidth and the highest frequency are the same value. However, consider a case where the signal is band-limited to a region from 30 MHz to 35 MHz. The Nyquist rate is twice the signal bandwidth, or 10 MS/s. If we simply used twice the highest frequency, we would think 70 MS/s would be necessary. If a high-resolution digitizer is used, say 14 bits per sample, then the data rate is 122.5 MB/s. With the lower sampling frequency of 10 MHz, the data rate is only 17.5 MB/s. Since the data rate a system can handle is always constrained, using a sampling rate that is very high will force the use of a lower-resolution converter to keep the data rate manageable. To prevent contamination of the signal, when undersampling a bandpass filter must be used to remove frequencies outside the band of interest.

Digital Down Conversion

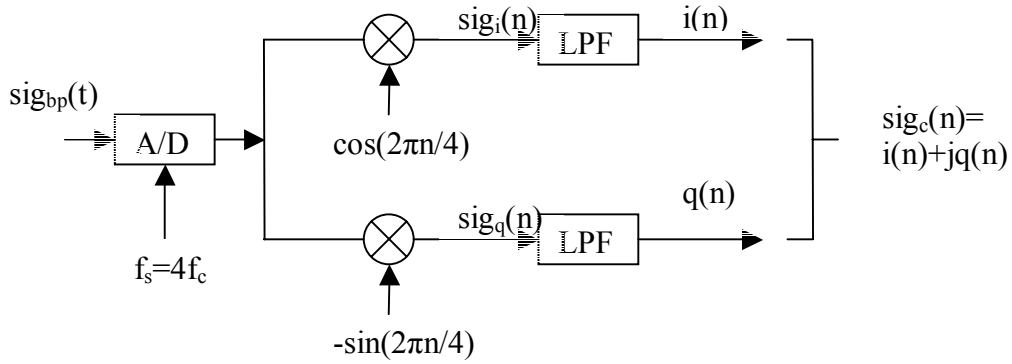


Quadrature Sampling Digital Down Converter

An extension of the idea of quadrature sampling is a quadrature digital down converter, shown in the Simulink model of Figure 5. For example, suppose that a signal with 5 MHz bandwidth centered at 50 MHz will be digitally recorded. To meet the Nyquist criterion, the signal would theoretically have to be sampled at a rate greater then 105 MS/s. However, in reality the sampling rate would be around 200 MS/s. Assuming a 12 bit ADC, the data generation rate is 300 MB/s. Collecting and storing data at this rate is a significant challenge. Usually, the important part of the signal is only the 5 MHz bandwidth. The idea behind digital down conversion is to only record the 5 MHz signal around 50 MHz, not the entire bandwidth. The down conversion process is made up of three steps: frequency shifting, sum-terms rejection by lowpass filtering, and downsampling.

There exists another method for performing complex down-conversion. It also involves mixing after digitization, but with the A/D sampling rate set to 4 times the

bandpass signal's center frequency. With this arrangement, the cosine and sine oscillator outputs are repetitive four-element sequences, 1,0,-1,0 for the cosine and 0,-1,0,1 for the sine.



Quadrature sampling with digital mixing method.

With this scheme, the sampling frequency on the digitizer appears to be at 4 times the center frequency of the input signal. Often, a rate this high would be unattractive. Using the technique of undersampling can help drop this value to something more practical. Consider a signal with an analog center frequency of 50 MHz. At first, it seems this signal must be sampled at 200 MHz to satisfy the scheme. However, if the sampling rate is chosen to be 40 MHz, the resulting aliasing can be of use. As discussed in the appendix, the signal will appear at 10 MHz, which is exactly one-fourth the sampling frequency.

Frequency Shifting

The frequency shifting property of the Fourier Transform states that

if

$h(t) \Leftrightarrow H(f)$

then

$h(t)e^{j2\pi f_0 t} \Leftrightarrow H(f - f_0)$

The multiplication of a signal by a sinusoid is described by the application of this frequency shifting theorem in conjunction with Euler's formulas:

$$\cos(2\pi f_0 t) = \frac{1}{2}(e^{j2\pi f_0 t} + e^{-j2\pi f_0 t})$$

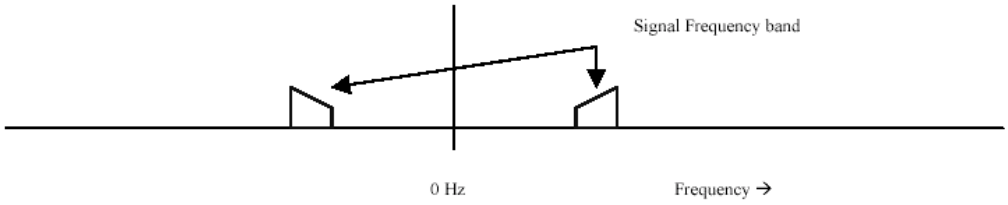
$$\sin(2\pi f_0 t) = \frac{1}{2j}(e^{j2\pi f_0 t} - e^{-j2\pi f_0 t})$$

Combining the two results yields:

$$h(t)\cos(2\pi f_0 t) \Leftrightarrow \frac{1}{2}[H(f - f_0) + H(f + f_0)]$$

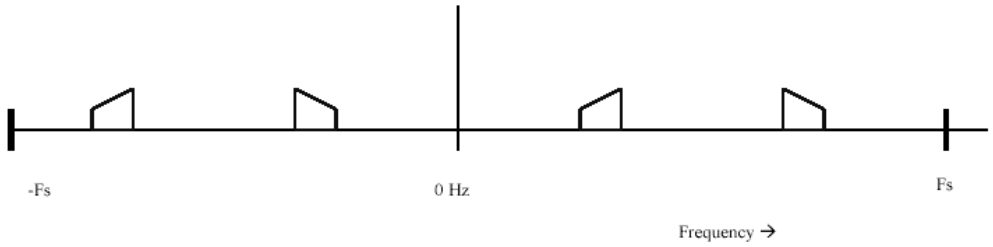
$$h(t)\sin(2\pi f_0 t) \Leftrightarrow \frac{1}{2j}[H(f - f_0) - H(f + f_0)]$$

The important information conveyed by these equations is that two frequency shifts occur to the spectrum of the modulating signal *f(t)*. It is desired that the center frequency of the original signal (50 MHz in this example) be shifted down to zero. Let the carrier frequency f_0 = 50 MHz. Now, the signal spectrum will shift down by 50 Mhz, and also up by 50 MHz. This is illustrated in the following figures.

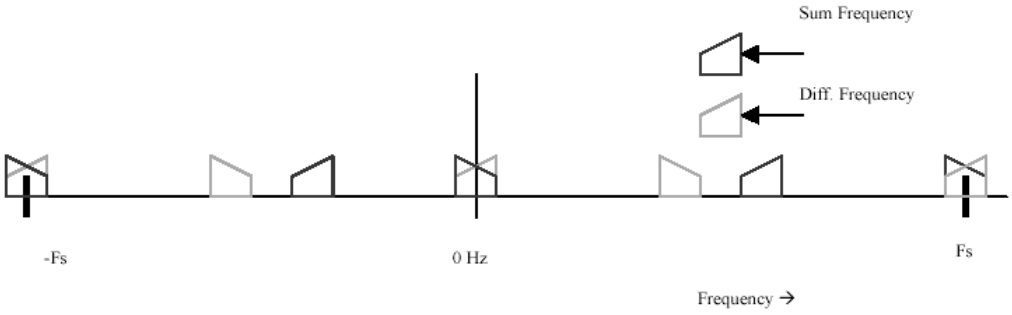


Spectrum of a continuous analog signal

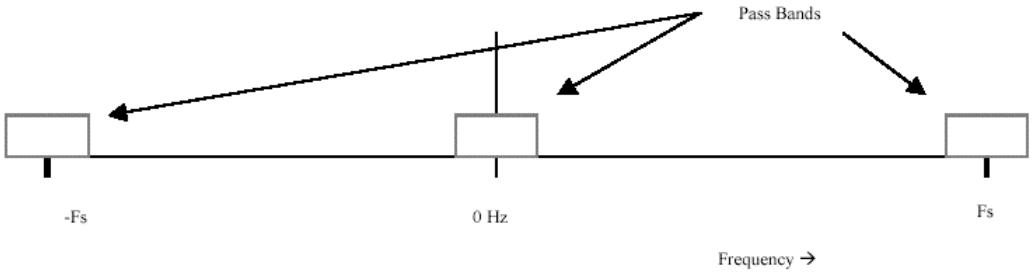
Sampling the signal results in periodic repetition of the spectrum, with the period equal to the sampling frequency.



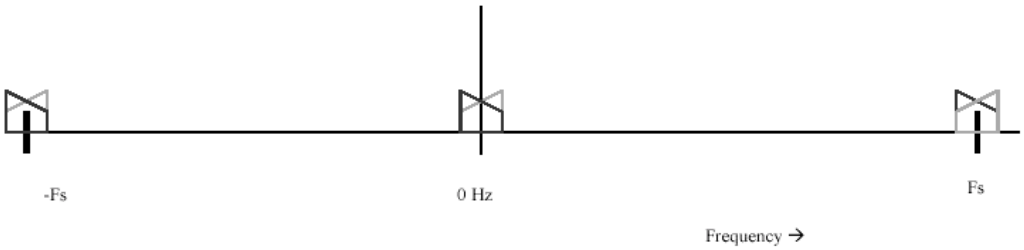
Spectrum of signal after sampling



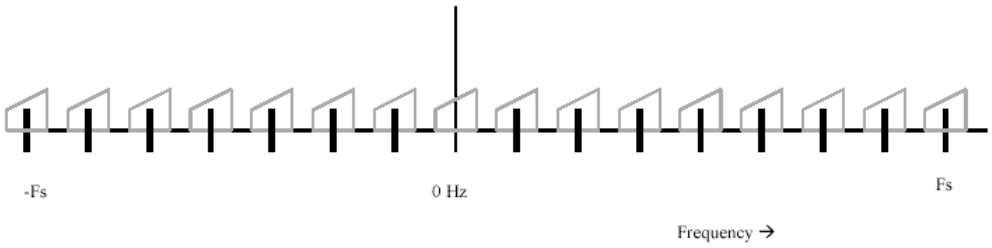
Spectrum of digitized signal after mixing



Lowpass digital filter frequency response



Spectrum of digitized signal after filtering



Spectrum of quadrature signal after downsampling

DFT Wrap-Around Order

The DFT of an *N*-point sequence is a complex *N*-point array. If the intersample period is chosen to be T, the sampling rate $f_s = 1/T$. Since the data samples are spaced evenly between 0 and the sampling frequency, the frequency bin spacing Δ is the sampling rate divided by the number of points, or $\Delta = f_s / N = 1/(NT)$.

n = 0 identifies the $f = 0$ frequency component whose value is equal to the average of all elements in the time waveform.

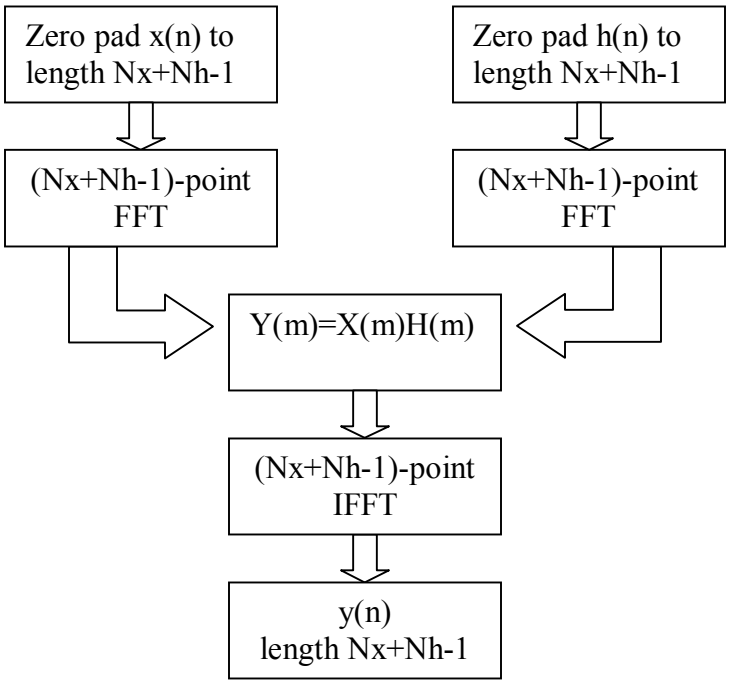
n = 1 to *N*/2 is mapped to the frequency $f_n = n\Delta = n / NT$, (all positive frequencies up to the Nyquist limit, $1/2T = f_s / 2$).

n = N − 1 down to N/2 is mapped to the negative frequency $f_n = (n - N)\Delta$, with *N*/2 sharing the negative Nyquist frequency and the positive Nyquist frequency component. This sharing comes from the periodic nature of the DFT:

$$f_n(n = N/2) = f_n(n = N/2 - N = -N/2)$$

The way to remember this scheme is that index zero is zero frequency, index *n* up to *N*/2 is positive frequencies, and, for index $n > N/2$, the index *n* labels the component value with negative frequency $(n - N)\Delta$. The component at index *n* has its corresponding negative-frequency partner at index $N - n$, which reverses the order of the negative frequency components.

The weights may be linearly convolved with the digitized time domain input vector. However, if the number of weights in the matched filter is increased, it becomes more efficient to perform fast convolution using an FFT. However, the application of convolution through the DFT must be done carefully. In the classic sense of a filtering operation, the convolution performed must be a linear type, not a circular type. The convolution that is effected through the use of the DFT is a circular convolution (Rabiner & Gold, 1975). To produce a linear convolution while using the DFT, zero-padding must take place in the input sequence, and in the impulse response, or weight vector, of the filter. This procedure is outlined in Figure X.



Block diagram of frequency domain convolution ("fast convolution")

References

[1] T.H. Stix, *Waves in Plasmas,* New York: American Institute of Physics, 1992.

[2] K.G. Budden, The Propogation of Radio Waves, Cambridge: Cambridge University Press, 1985.

[3] K.G. Budden, Radio Waves in the Ionosphere, Cambridge: Cambridge University Press, 1961.

[4] K. Davies, *Ionospheric Radio,* New York: Peter Peregrinus Ltd., 1990.

[5] J.S. Abel and J.O. Smith, "Robust Design of Very High-Order Allpass Dispersion Filters," in *Proc. 9th Int. Conf. on Digital Audio Effects*, Montreal, Canada, Sept. 2006.

[6] H.W. Schuessler and P. Steffen, "On the Design of Allpasses with Prescribed Group Delay," *Int. Conf. on Acousics and Speech Signal Processing*, pp. 1313-1316, 1990.

[7] B. Yegnanarayana, "Design of Recursive Group-Delay Filters by Autoregressive Modeling", *IEEE Trans. On Acoustics, Speech and Signal Processing,* vol. ASSP-30, no. 4, August 1982.

[8] R. Rajagopal and L . Wenzel, "Peak Locations in All-Pass Signals: The Makhoul Conjecture Challenge," *IEE Signal Processing Magazine*, vol. 17, no. 3, pp. 8-11, May 2000.

[9] R.E. Ziemer and W.H. Tranter, *Principles of Communications*, 4th ed., New York: John Wiley & Sons, 1995.

[10] A. Antoniou, *Digital Filters: Analysis and Design*, New York: McGraw-Hill Book Co., 1979.

[11] A. Antoniou, *Digital Signal Processing: Signals, Systems, and Filters*, McGraw-Hill, New York, 2006.

[12] R. Lyons, *Understanding Digital Signal Processing*, Prentice Hall, Upper Saddle River, 2004.

[13] W. Breiland. (2006, October 20). *Simple Model for Pulse Propagation Through the Ionosphere* [Monograph], Albuquerque.

[14] A.V. Oppenheim and R.W. Schafer, *Discrete-Time Signal Processing*, Prentice-Hall, Upper Saddle River, 1999.

[15] S.D. Stearns, *Digital Signal Processing with Examples in MATLAB*, CRC Press, New York, 2003.

[16] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer, New York, 1999.

[17] E.K.P Chong and S.H. Zak, *An Introduction to Optimization*, Wiley, New York, 2004.

[18] J.E. Dennis, Jr., and R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, 1996.

[19] N. Sherif, "On the Computation of a Matrix Inverse Square Root", *Journal on Computing*, vol. 46, no. 4, p. 295-305, Springer Wein, New York, 1991.